End-to-End AI Systems Adversarial Training

Yann Le Cun
Facebook AI Research,
Center for Data Science, NYU
Courant Institute of Mathematical Sciences, NYU
http://yann.lecun.com

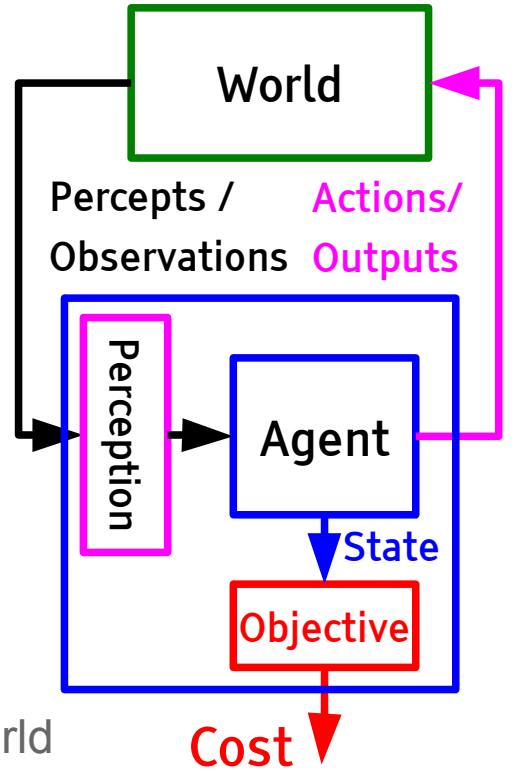
Obstacles to AI

How could machines acquire common sense?

- TL;DR:
  - Machines don’t have common sense
  - To acquire it, they must learn how the world works by observation
  - Predictive/unsupervised learning is the missing link

Obstacles to Progress in AI

- **Machines need to learn/understand how the world works**
  - Physical world, digital world, people,….
  - They need to acquire some level of common sense
- **They need to learn a very large amount of background knowledge**
  - Through observation and action
- **Machines need to perceive the state of the world**
  - So as to make accurate predictions and planning
- **Machines need to update and remember world state estimates**
  - Paying attention to important events. Remember relevant events
- **Machines neet to reason and plan**
  - Predict which sequence of actions will lead to a desired state of the world



- **Intelligence & Common Sense =**
  Perception + Predictive Model + Memory + Reasoning & Planning

What is Common Sense?

- **“The trophy doesn’t fit in the suitcase because it’s too large/small”**
  - (winograd schema)

- **“Tom picked up his bag and left the room”**

- **We have common sense because we know how the world works**

- **How do we get machines to learn that?**

Common Sense is the ability to fill in the blanks

- Infer the state of the world from partial information
- Infer the future from the past and present
- Infer past events from the present state

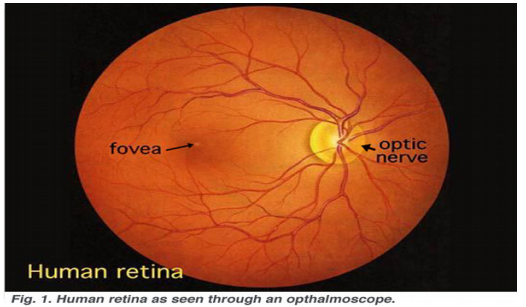
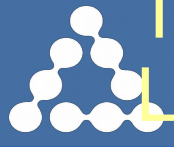


Fig. 1. Human retina as seen through an opthalmoscope.

- Filling in the visual field at the retinal blind spot
- Filling in occluded images
- Fillling in missing segments in text, missing words in speech.
- Predicting the consequences of our actions
- Predicting the sequence of actions leading to a result

- **Predicting any part of the past, present or future percepts from whatever information is available.**

- That's what **predictive learning** is
- But really, that's what many people mean by unsupervised learning



The Necessity of Unsupervised Learning / Predictive Learning

- **The number of samples required to train a large learning machine (for any task) depends on the amount of information that we ask it to predict.**
  - The more you ask of the machine, the larger it can be.

- **"The brain has about 10^14 synapses and we only live for about 10^9 seconds. So we have a lot more parameters than data. This motivates the idea that we must do a lot of unsupervised learning since the perceptual input (including proprioception) is the only place we can get 10^5 dimensions of constraint per second."**
  - Geoffrey Hinton (in his 2014 AMA on Reddit)
  - (but he has been saying that since the late 1970s)

- **Predicting human-provided labels is not enough**

- **Predicting a value function is not enough**

How Much Information Does the Machine Need to Predict?

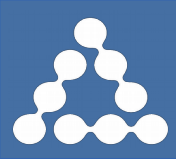
- **"Pure" Reinforcement Learning (cherry)**
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- **Supervised Learning (icing)**
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- **Unsupervised/Predictive Learning (cake)**
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**

- **(Yes, I know, this picture is slightly offensive to RL folks. But I'll make it up)**

VizDoom Champion: Actor-Critic RL system from FAIR

Winner of the VizDoom 2016 competition.

[Wu & Tian, ICLR 2017]



“StarCraft: Brood War” RL-based system for battle tactics

Plug: TorchCraft: interface between Torch and StarCraft (on github)

[Usunier, Synnaeve, Lin, Chintala, ICLR 2017]

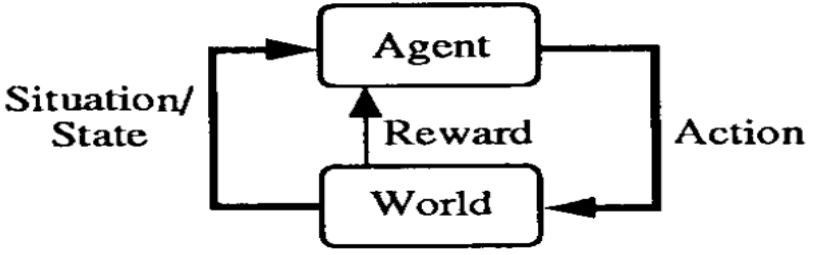
Sutton's Dyna Architecture: "try things in your head before acting"

- **Dyna: an Integrated Architecture for Learning, Planning and Reacting**
  - [Rich Sutton, ACM S

The main idea of Dyna is the old, commonsense idea that planning is 'trying things in your head,' using an internal model of the world (Craik, 1943; Dennett, 1978; Sutton & Barto, 1981). This suggests the existence of a more primitive process for trying things *not* in your head, but through direct interaction with the world. *Reinforcement learning* is the name we use for this more primitive, direct kind of trying, and Dyna is the extension of reinforcement learning to include a learned world model.

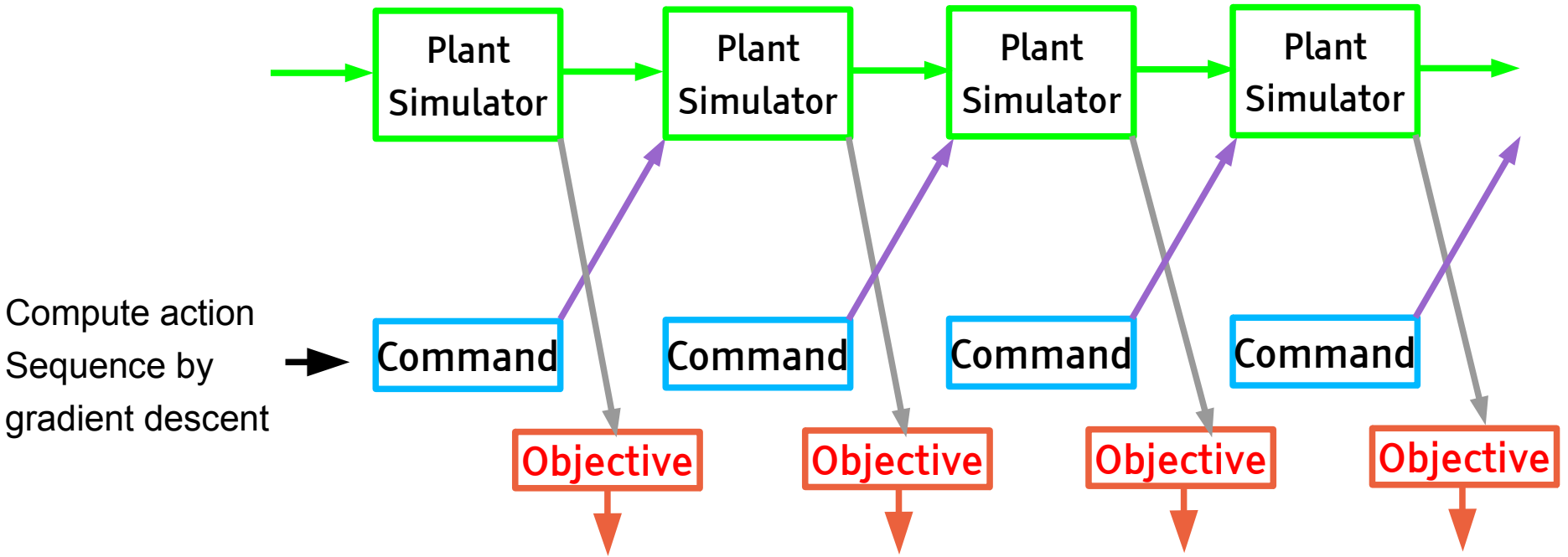
REPEAT FOREVER:

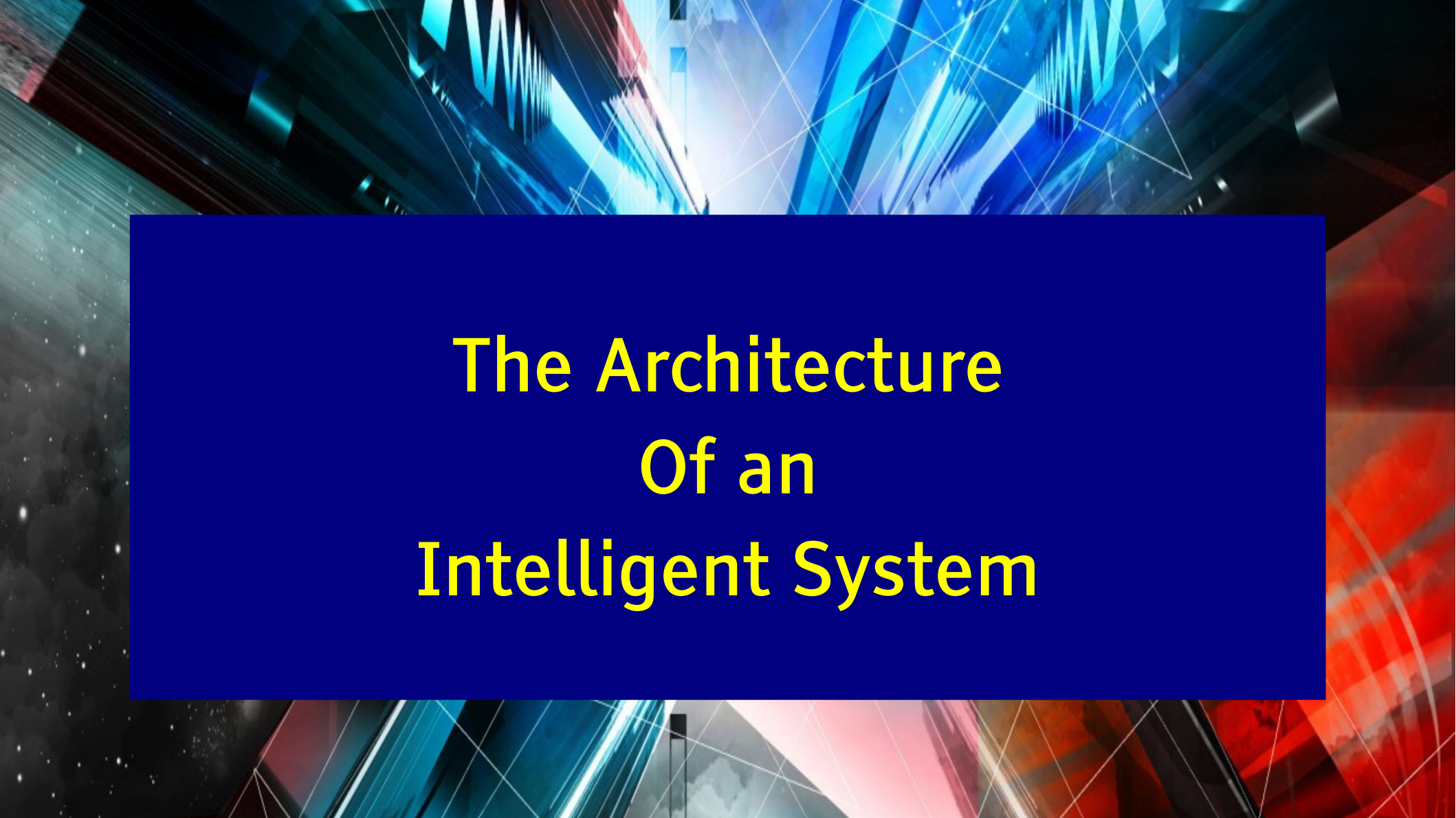
1. Observe the world's state and reactively choose an action based on it;
2. Observe resultant reward and new state;
3. Apply reinforcement learning to this experience;
4. Update action model based on this experience;
5. Repeat K times:
   - 5.1 Choose a hypothetical world state and action;
   - 5.2 Predict resultant reward and new state using action model;
   - 5.3 Apply reinforcement learning to this hypothetical experience.



Classical model-based optimal control

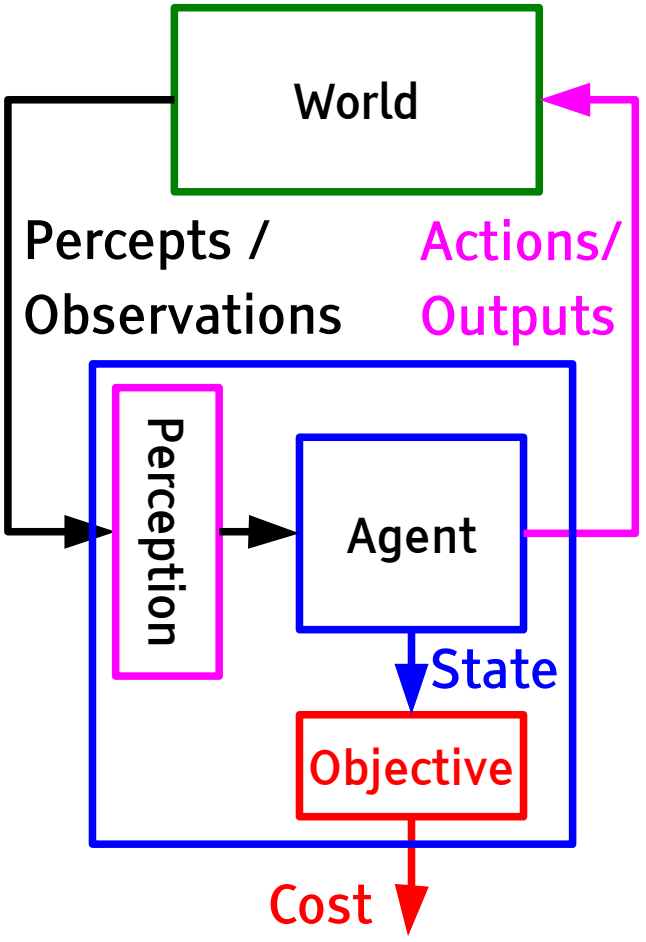
- **Simulate the world (the plant) with an initial control sequence**
- **Adjust the control sequence to optimize the objective through gradient descent**
- **Backprop through time was invented by control theorists in the late 1950s**
  - it's sometimes called the adjoint state method
  - [Athans & Falb 1966, Bryson & Ho 1969]




The Architecture
Of an
Intelligent System

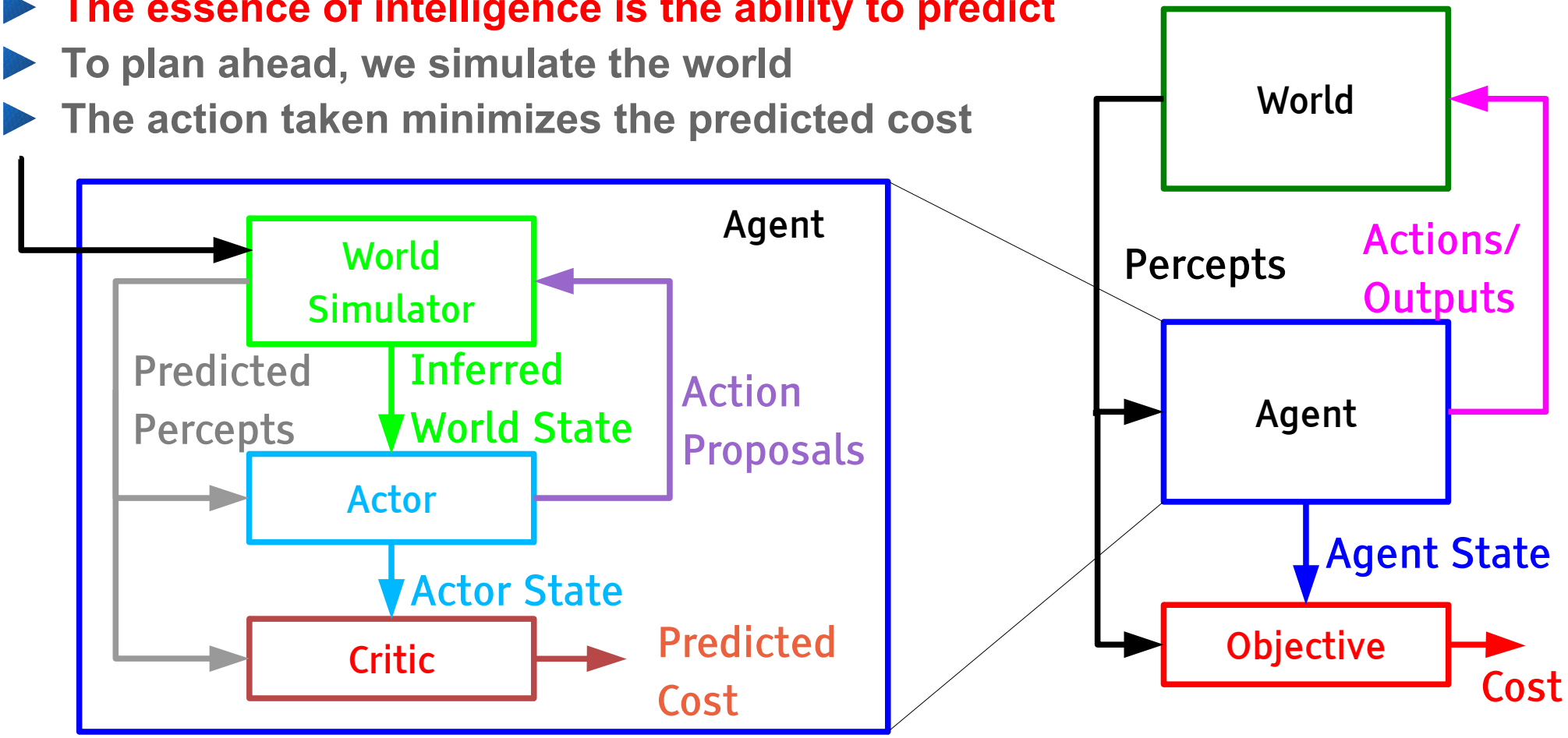
The Architecture of an AI system

- **Perception:** estimates the state of the world
  - Vision, Speech, Audio
- **Agent**
  - Prediction, Causal Inference
  - Planning, Reasoning, Working Memory
- **Objective:** Measures the agent's "happiness"
  - Agent acts to optimize the objective
  - Drives the system to do what we want
  - Parts of it are hardwired and immutable
  - Parts of it is trainable



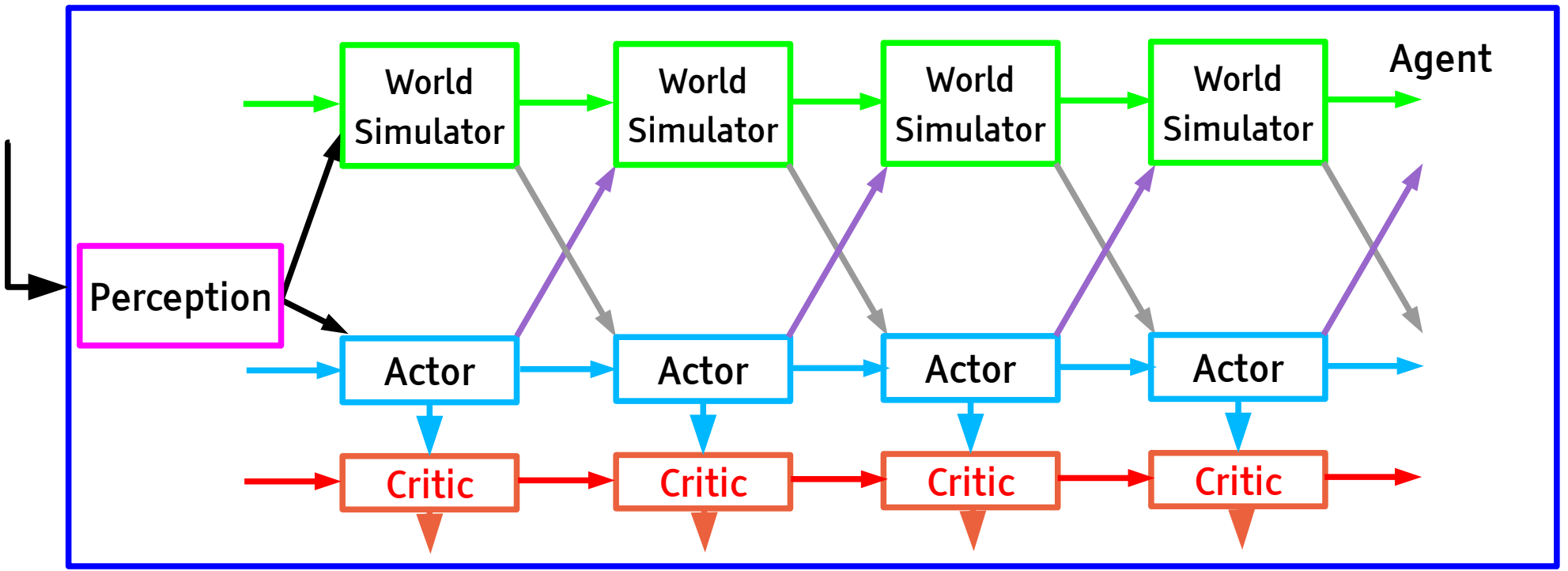
AI System: Predicting + Planning = Reasoning

- **The essence of intelligence is the ability to predict**
- To plan ahead, we simulate the world
- The action taken minimizes the predicted cost



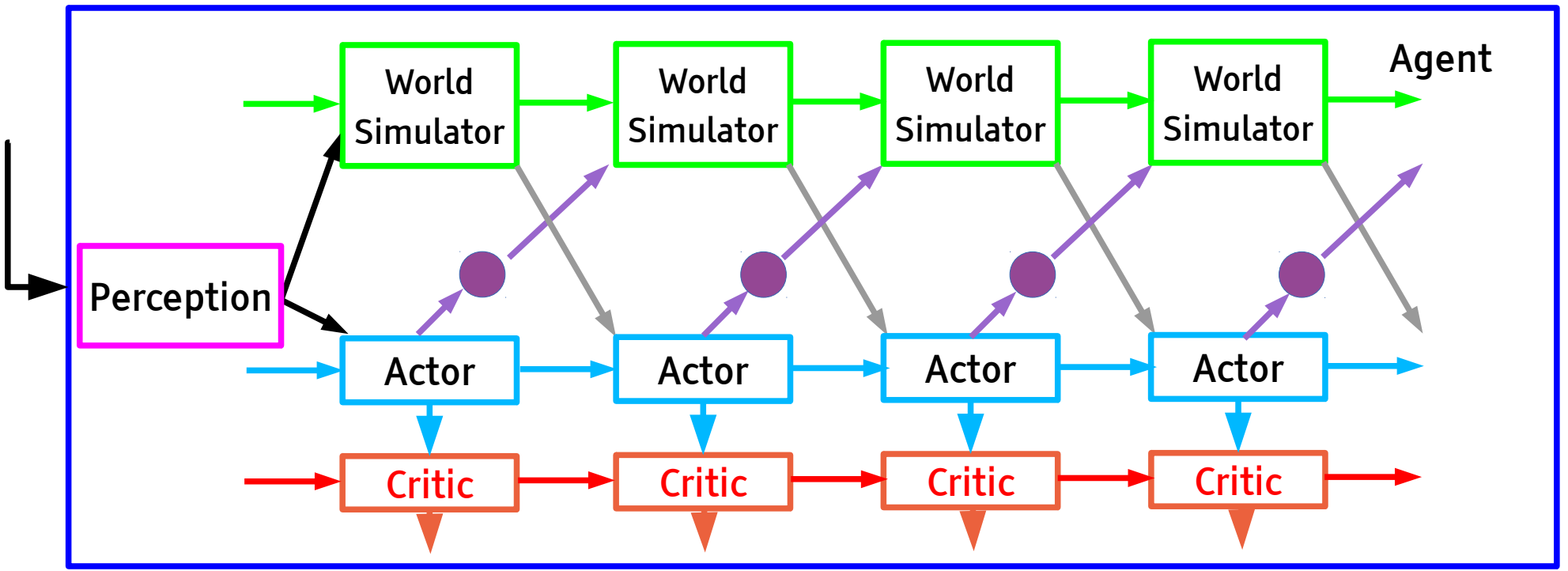
What we need is Model-Based Reinforcement Learning

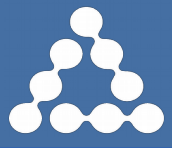
- The essence of intelligence is the ability to predict
- To plan ahead, we must simulate the world, so as to minimizes the predicted value of some objective function.



Training the Actor with Optimized Action Sequences

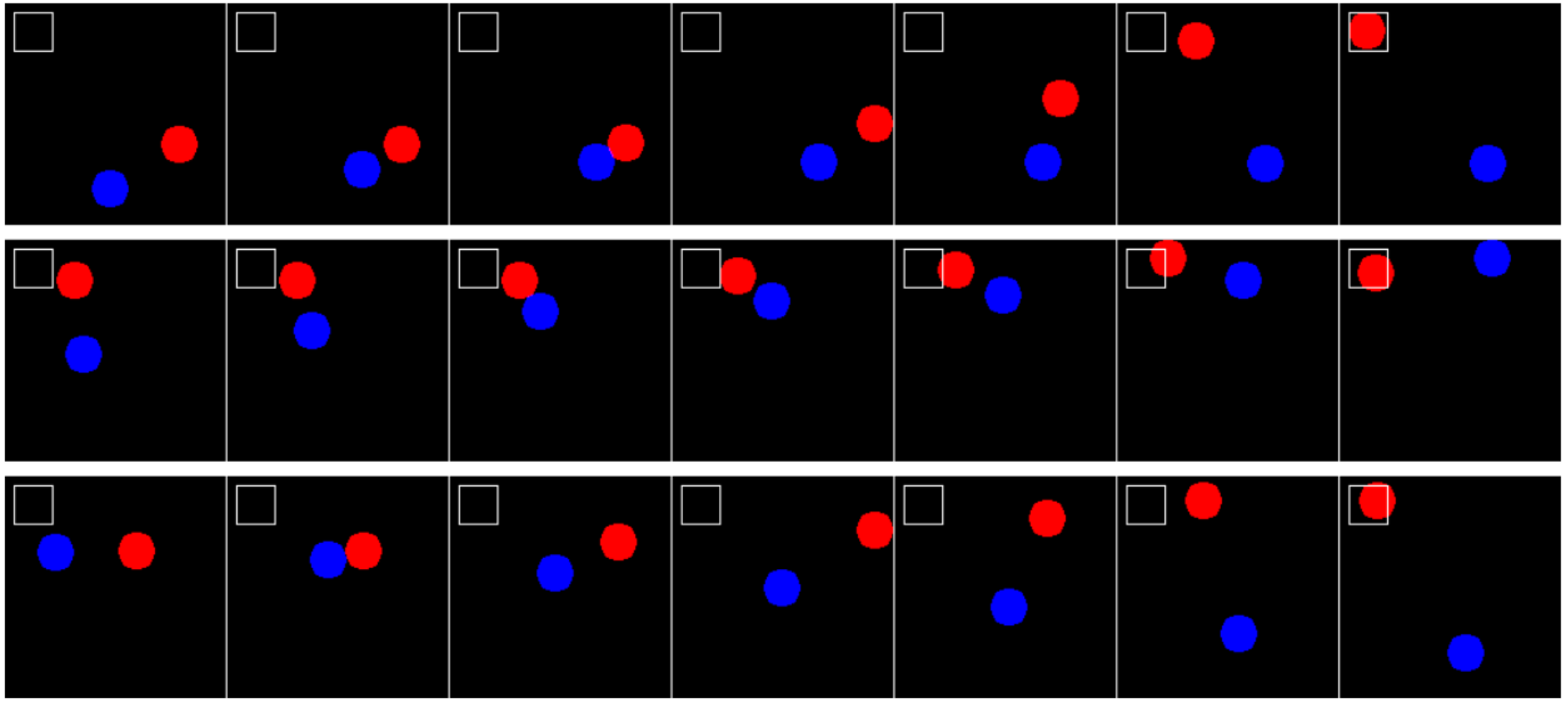
- **1. Find action sequence through optimization**
- **2. Use sequence as target to train the actor**
  - Over time we get a compact policy that requires no run-time optimization

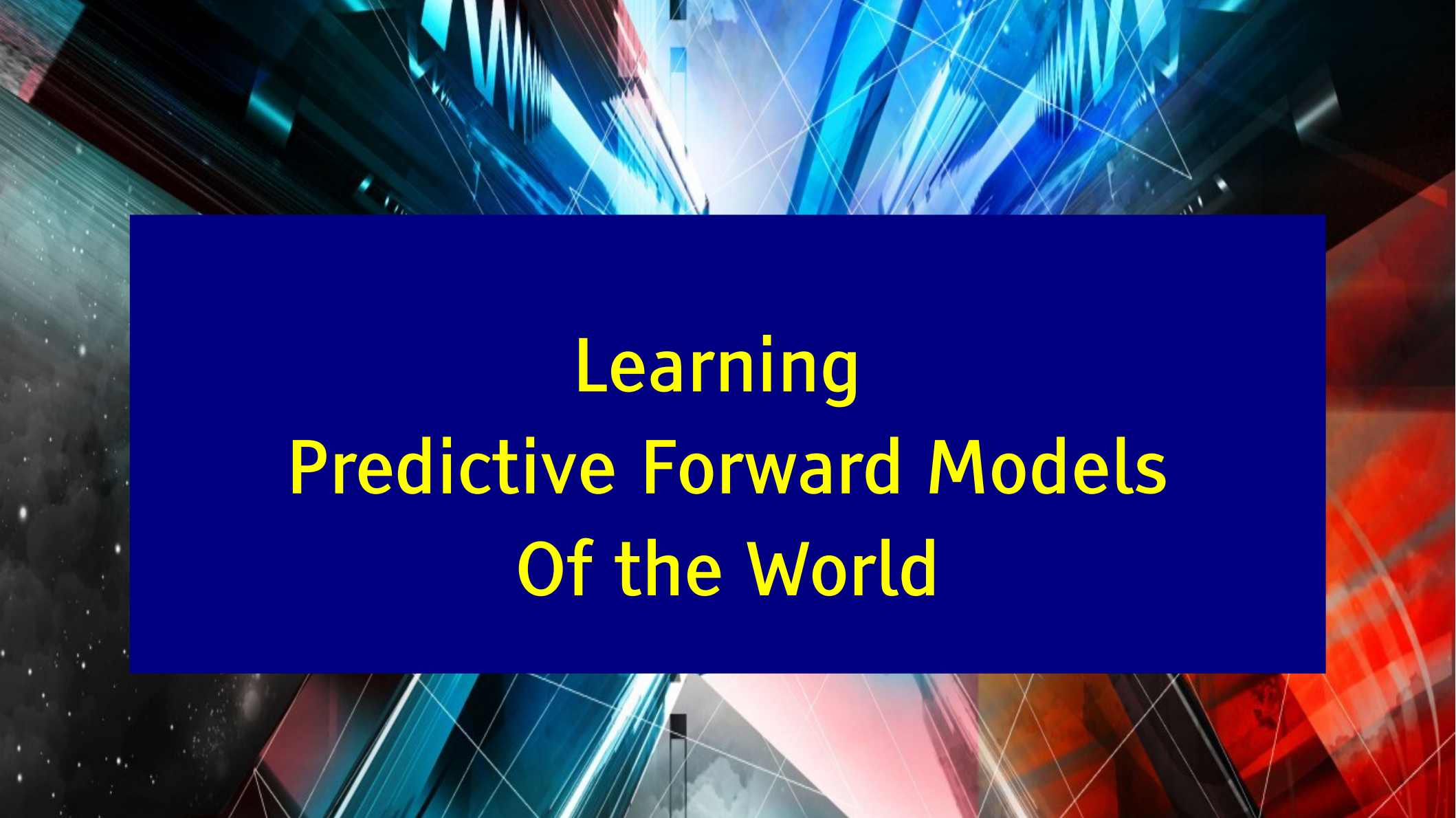




Learned model of Billiard + Planning

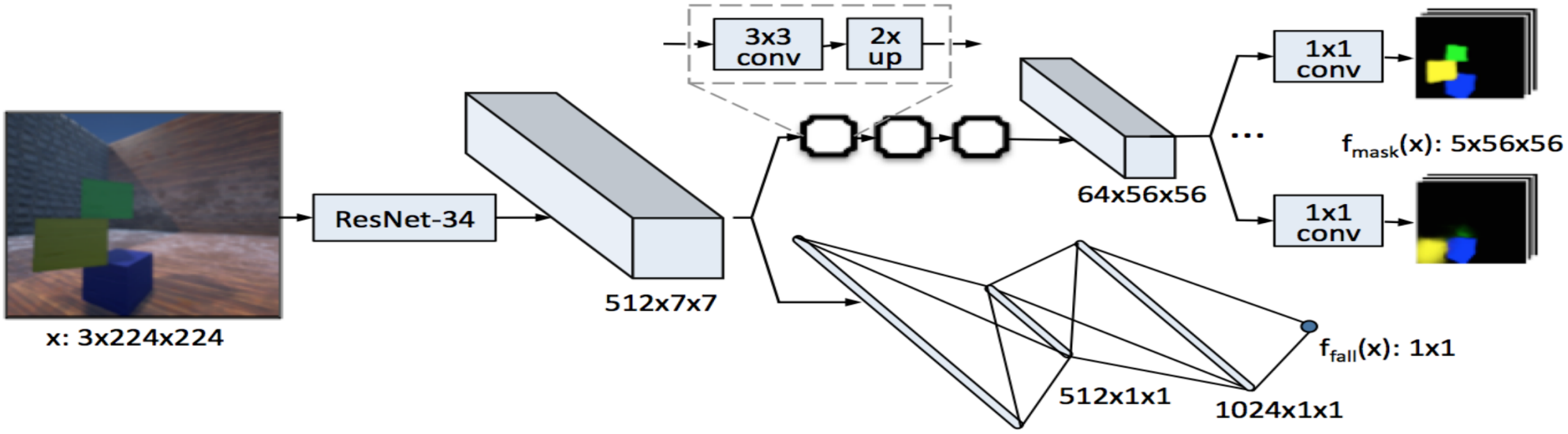
- **Forward model: Entity RNN + Interactions** [Henaff, Whitney, LeCun 2017]




Learning
Predictive Forward Models
Of the World

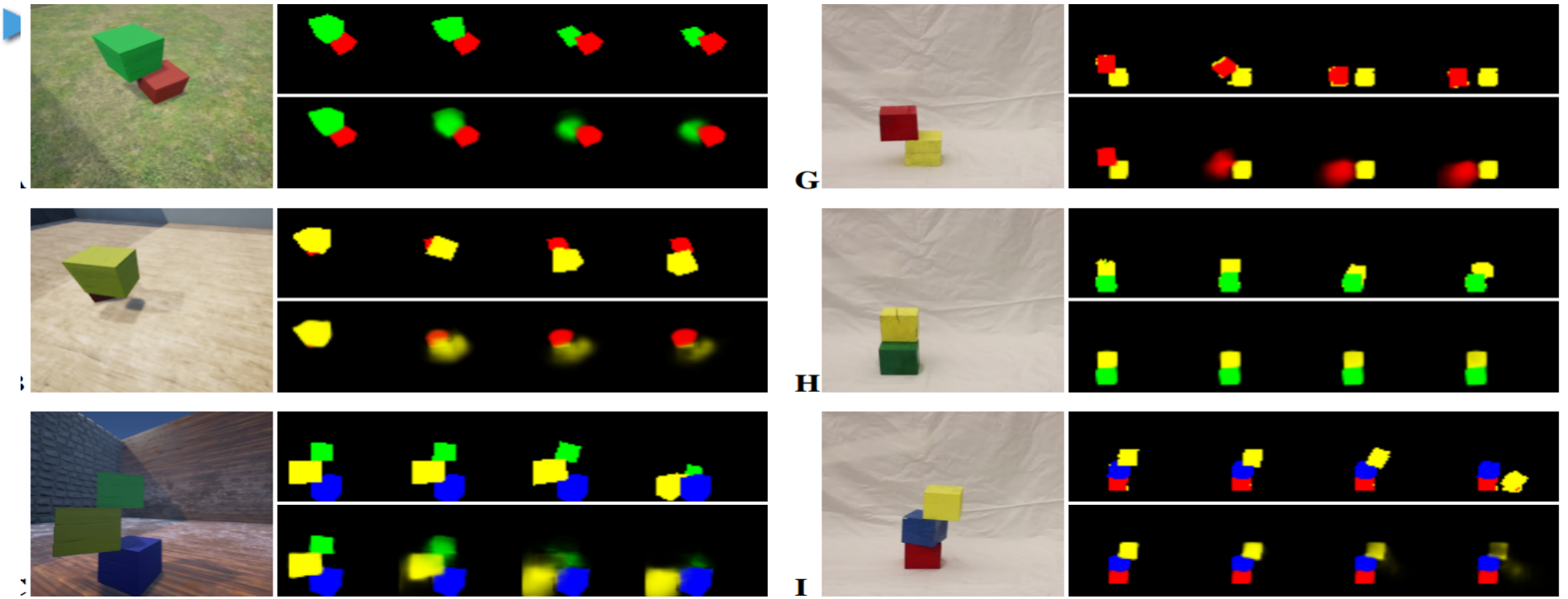
Learning Physics (PhysNet)

- **[Lerer, Gross, Fergus arxiv:1603.01312]**
  - ConvNet produces object masks that predict the trajectories of falling blocks
  - Uses the Unreal game engine.

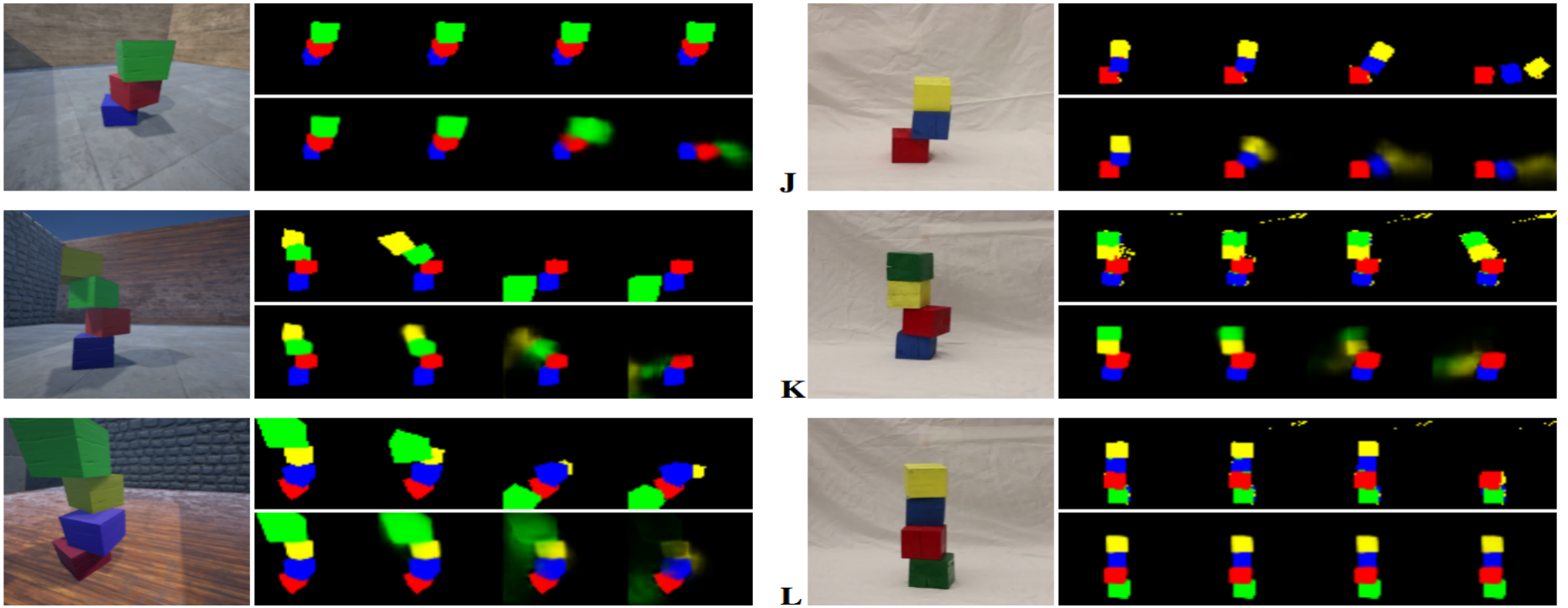


Learning Physics (PhysNet)

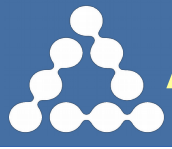
- **[Lerer, Gross, Fergus arxiv:1603.01312]**
  - ConvNet produces object masks that predict the trajectories of falling blocks



Learning Physics (PhysNet)

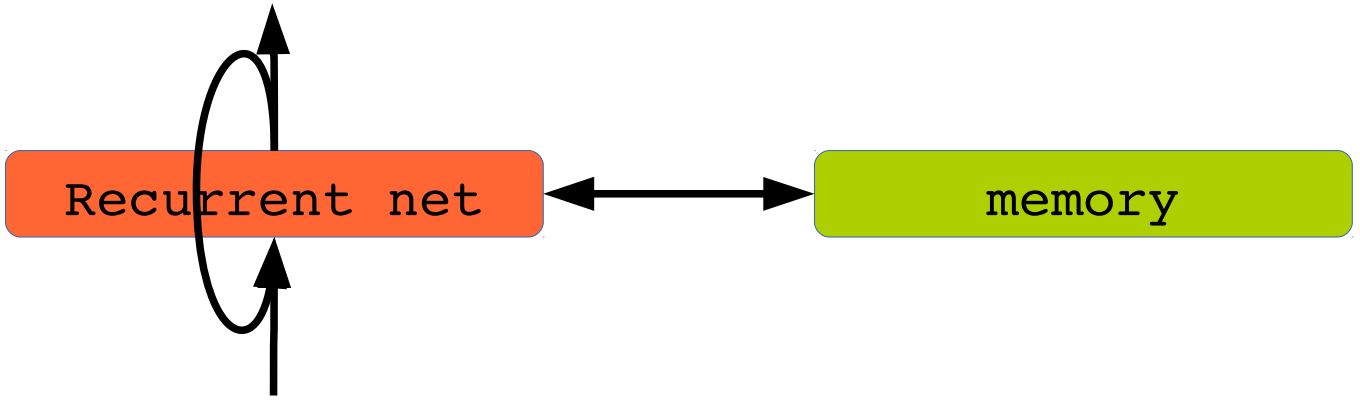
- **[Lerer, Gross, Fergus arxiv:1603.01312]**
  - ConvNet produces object masks that predict the trajectories of falling blocks
  - 


Inferring the
State of the World from Text:
Entity RNN



Augmenting Neural Nets with a Memory Module

- **Recurrent networks cannot remember things for very long**
  - The cortex only remember things for 20 seconds
- **We need a "hippocampus" (a separate memory module)**
  - LSTM [Hochreiter 1997], registers
  - **Memory networks** [Weston et 2014] (FAIR), associative memory
  - **Stacked-Augmented Recurrent Neural Net** [Joulin & Mikolov 2014] (FAIR)
  - **Neural Turing Machine** [Graves 2014],
  - **Differentiable Neural Computer** [Graves 2016]

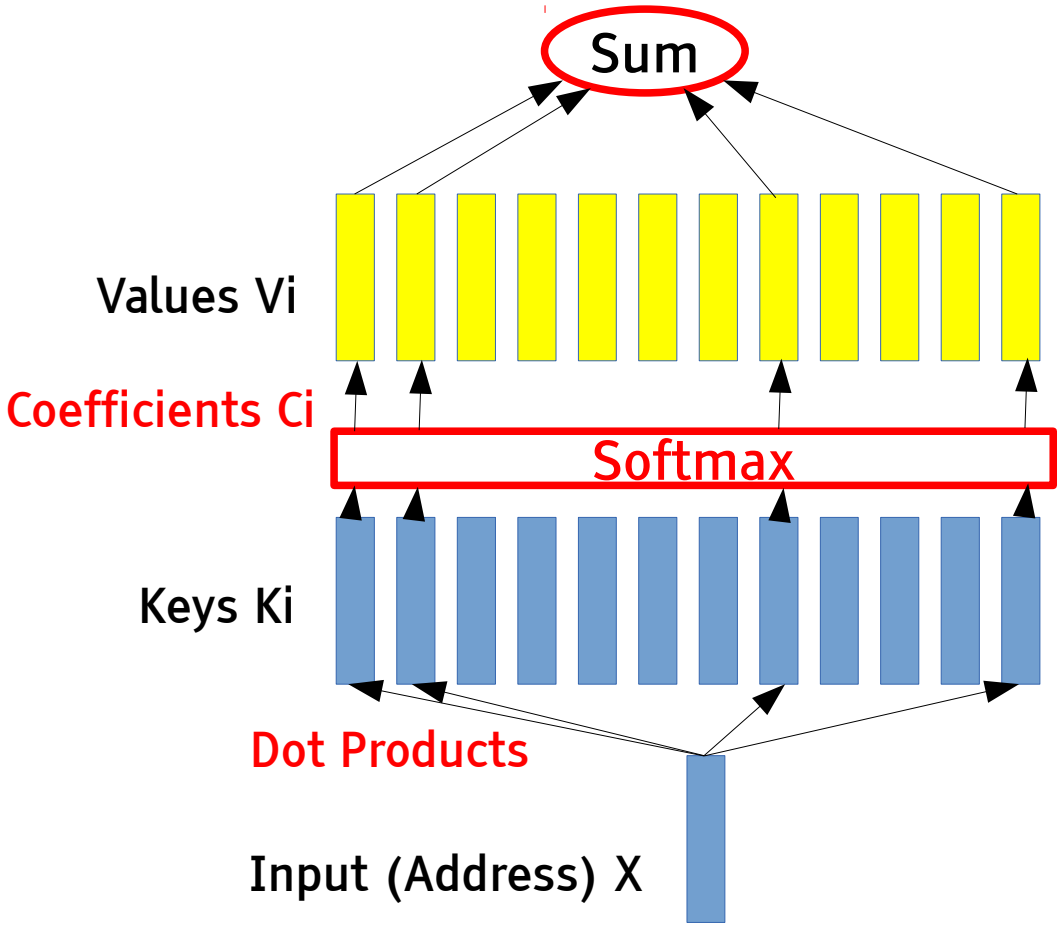


Differentiable Memory

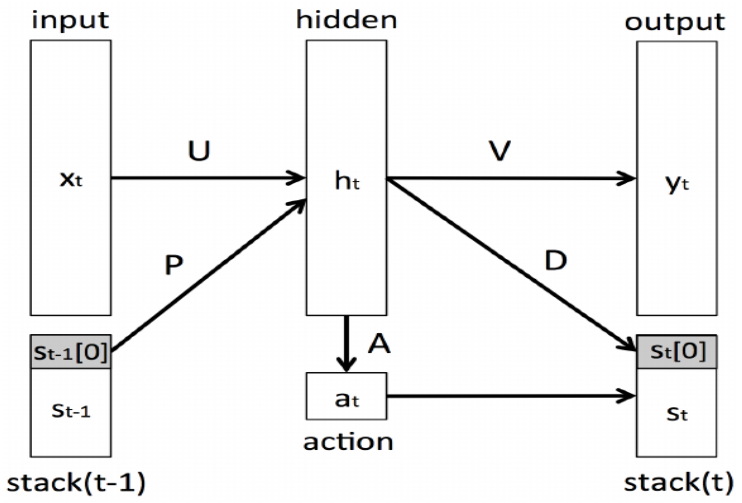
- Like a “soft” RAM circuit
- Or a “soft” hash table
- Stores Key-Value pairs (Ki,Vi)

$$C_i = \frac{e^{K_i^T X}}{\sum_j e^{K_j^T X}}$$

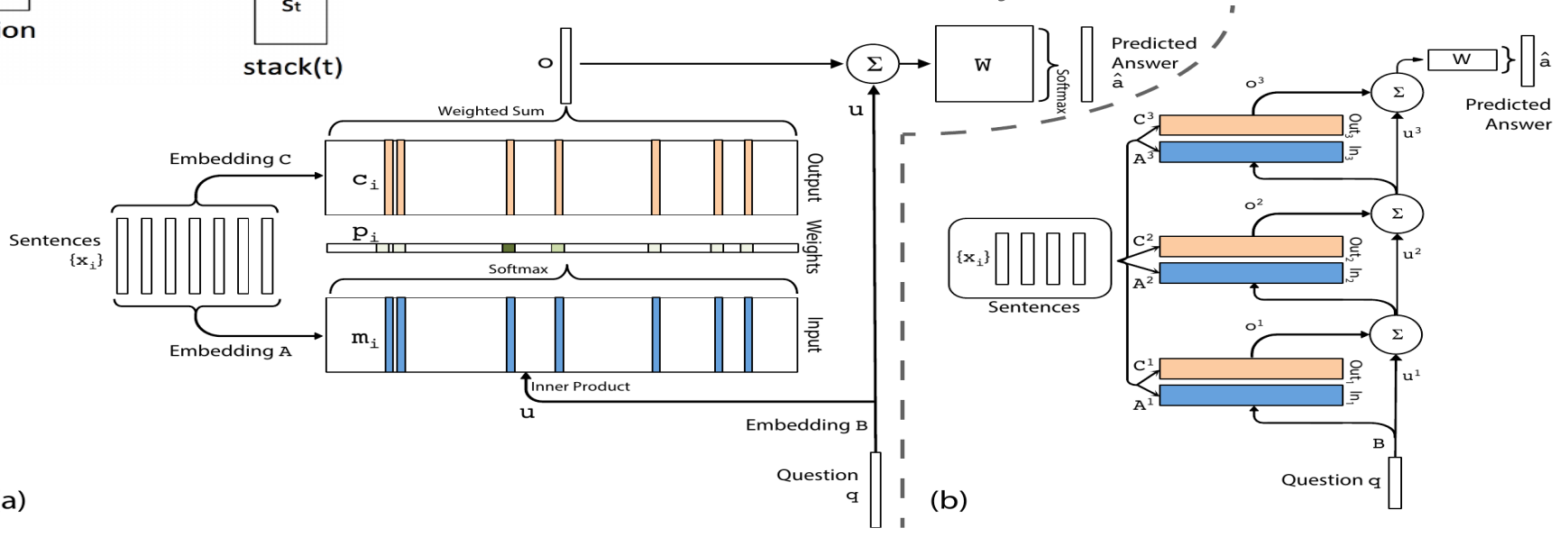
$$Y = \sum_i C_i V_i$$



Memory/Stack-Augmented Recurrent Nets



- **[Joulin & Mikolov, ArXiv:1503.01007]**
  - Stack-augmented RNN
- **[Sukhbataar, Szlam, Weston, Fergus NIPS 2015]**
  - ArXiv:1503.08895]
- **Weakly-supervised MemNN:**
  - discovers which memory location to use.



Memory Network [Weston, Chopra, Bordes 2014]

Add a short-term memory to a network

http://arxiv.org/abs/1410.3916

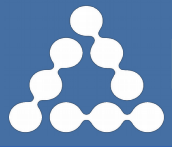
I: (input feature map) – converts the incoming input to the internal feature representation.
G: (generalization) – updates old memories given the new input.
O: (output feature map) – produces a new output (in the feature representation space), given the new input and the current memory.
R: (response) – converts the output into the response format desired. For example, a textual response or an action.

Method	F1
(Fader et al., 2013) [4]	0.54
(Bordes et al., 2014) [3]	0.73
MemNN	0.71
MemNN (with BoW features)	0.79

Results on Question Answering Task

Bilbo travelled to the cave.
Gollum dropped the ring there.
Bilbo took the ring.
Bilbo went back to the Shire.
Bilbo left the ring there.
Frodo got the ring.
Frodo journeyed to Mount-Doom.
Frodo dropped the ring there.
Sauron died.
Frodo went back to the Shire.
Bilbo travelled to the Grey-havens.
The End.
Where is the ring? A: Mount-Doom
Where is Bilbo now? A: Grey-havens
Where is Frodo now? A: Shire

Fig. 2. An example story with questions correctly answered by a MemNN. The MemNN was trained on the simulation described in Section 4.2 and had never seen many of these words before, e.g. Bilbo, Frodo and Gollum.



End-to-End Memory Network on bAbI tasks [Weston 2015]

```
Sam walks into the kitchen.
Sam picks up an apple.
Sam walks into the bedroom.
Sam drops the apple.
Q: Where is the apple?
A. Bedroom
```

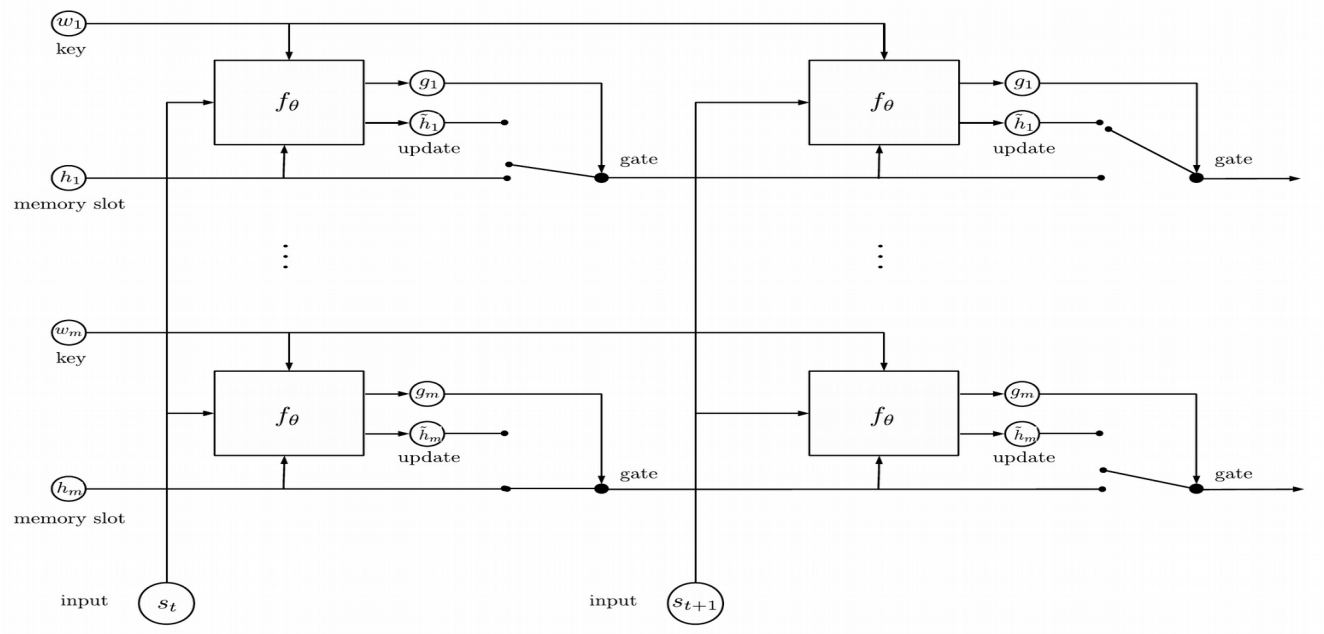
```
Brian is a lion.
Julius is a lion.
Julius is white.
Bernhard is green.
Q: What color is Brian?
A. White
```

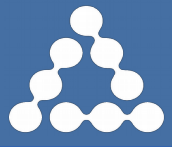
```
Mary journeyed to the den.
Mary went back to the kitchen.
John journeyed to the bedroom.
Mary discarded the milk.
Q: Where was the milk before the den?
A. Hallway
```

	Baseline			MemN2N								
Task	Strongly Supervised MemNN [21]	LSTM [21]	MemNN WSH	BoW	PE	PE LS	PE LS RN	1 hop PE LS joint	2 hops PE LS joint	3 hops PE LS joint	PE LS RN joint	PE LS LW joint
1: 1 supporting fact	0.0	50.0	0.1	0.6	0.1	0.2	0.0	0.8	0.0	0.1	0.0	0.1
2: 2 supporting facts	0.0	80.0	42.8	17.6	21.6	12.8	8.3	62.0	15.6	14.0	11.4	18.8
3: 3 supporting facts	0.0	80.0	76.4	71.0	64.2	58.8	40.3	76.9	31.6	33.1	21.9	31.7
4: 2 argument relations	0.0	39.0	40.3	32.0	3.8	11.6	2.8	22.8	2.2	5.7	13.4	17.5
5: 3 argument relations	2.0	30.0	16.3	18.3	14.1	15.7	13.1	11.0	13.4	14.8	14.4	12.9
6: yes/no questions	0.0	52.0	51.0	8.7	7.9	8.7	7.6	7.2	2.3	3.3	2.8	2.0
7: counting	15.0	51.0	36.1	23.5	21.6	20.3	17.3	15.9	25.4	17.9	18.3	10.1
8: lists/sets	9.0	55.0	37.8	11.4	12.6	12.7	10.0	13.2	11.7	10.1	9.3	6.1
9: simple negation	0.0	36.0	35.9	21.1	23.3	17.0	13.2	5.1	2.0	3.1	1.9	1.5
10: indefinite knowledge	2.0	56.0	68.7	22.8	17.4	18.6	15.1	10.6	5.0	6.6	6.5	2.6
11: basic coreference	0.0	38.0	30.0	4.1	4.3	0.0	0.9	8.4	1.2	0.9	0.3	3.3
12: conjunction	0.0	26.0	10.1	0.3	0.3	0.1	0.2	0.4	0.0	0.3	0.1	0.0
13: compound coreference	0.0	6.0	19.7	10.5	9.9	0.3	0.4	6.3	0.2	1.4	0.2	0.5
14: time reasoning	1.0	73.0	18.3	1.3	1.8	2.0	1.7	36.9	8.1	8.2	6.9	2.0
15: basic deduction	0.0	79.0	64.8	24.3	0.0	0.0	0.0	46.4	0.5	0.0	0.0	1.8
16: basic induction	0.0	77.0	50.5	52.0	52.1	1.6	1.3	47.4	51.3	3.5	2.7	51.0
17: positional reasoning	35.0	49.0	50.9	45.4	50.1	49.0	51.0	44.4	41.2	44.5	40.4	42.6
18: size reasoning	5.0	48.0	51.3	48.1	13.6	10.1	11.1	9.6	10.3	9.2	9.4	9.2
19: path finding	64.0	92.0	100.0	89.7	87.4	85.6	82.8	90.7	89.9	90.2	88.0	90.6
20: agent's motivation	0.0	9.0	3.6	0.1	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.2
Mean error (%)	6.7	51.3	40.2	25.1	20.3	16.3	13.9	25.8	15.6	13.3	12.4	15.2
Failed tasks (err. $>$ 5%)	4	20	18	15	13	12	11	17	11	11	11	10
On 10k training data												
Mean error (%)	3.2	36.4	39.2	15.4	9.4	7.2	6.6	24.5	10.9	7.9	7.5	11.0
Failed tasks (err. $>$ 5%)	2	16	17	9	6	4	4	16	7	6	6	6

Entity Recurrent Neural Net

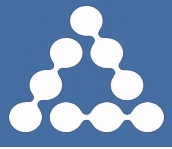
- Maintains a current estimate of the state of the world.
- Each module is a recurrent net with a “memory”
- Each input event causes some of the memory cells to get updated
- **“Tracking the World State with Recurrent Entity Networks”, Henaff, Weston, Szlam, Bordes, LeCun, ICLR 2017**





EntNet is the first model to solve all 20 bAbI tasks

Task	D-NTM	MemN2N	DNC	DMN+	EntNet
1: 1 supporting fact	4.4	0	0	0	0
2: 2 supporting facts	27.5	0.3	0.4	0.3	0.1
3: 3 supporting facts	71.3	2.1	1.8	1.1	4.1
4: 2 argument relations	0	0	0	0	0
5: 3 argument relations	1.7	0.8	0.8	0.5	0.3
6: yes/no questions	1.5	0.1	0	0	0.2
7: counting	6.0	2.0	0.6	2.4	0
8: lists/sets	1.7	0.9	0.3	0.0	0.5
9: simple negation	0.6	0.3	0.2	0.0	0.1
10: indefinite knowledge	19.8	0	0.2	0	0.6
11: basic coreference	0	0.0	0	0.0	0.3
12: conjunction	6.2	0	0	0.2	0
13: compound coreference	7.5	0	0	0	1.3
14: time reasoning	17.5	0.2	0.4	0.2	0
15: basic deduction	0	0	0	0	0
16: basic induction	49.6	51.8	55.1	45.3	0.2
17: positional reasoning	1.2	18.6	12.0	4.2	0.5
18: size reasoning	0.2	5.3	0.8	2.1	0.3
19: path finding	39.5	2.3	3.9	0.0	2.3
20: agent's motivation	0	0	0	0	0
Failed Tasks (> 5% error):	9	3	2	1	**0**
Mean Error:	12.8	4.2	3.8	2.8	**0.5**



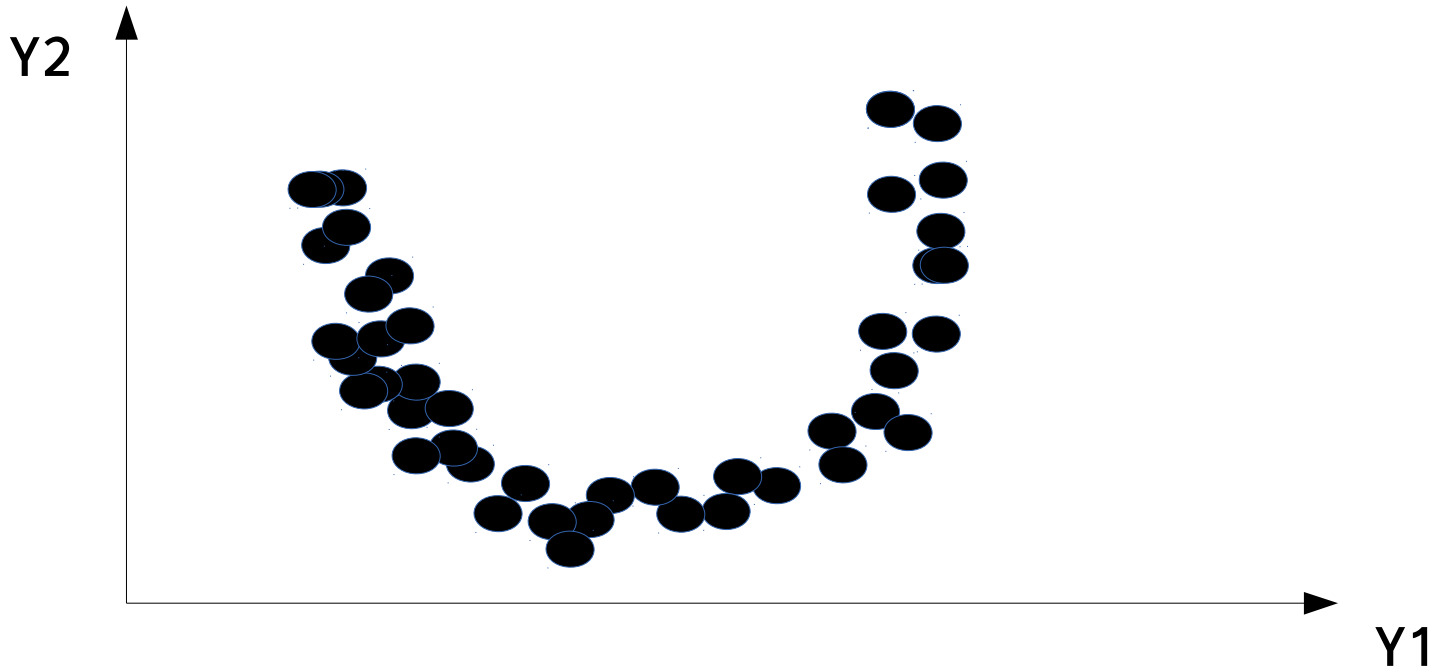
EntNet is the first model to solve all 20 bAbI tasks

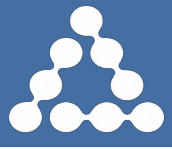
Story	Key	1-NN	2-NN
mary got the milk there	football	hallway (0.135)	dropped (0.056)
john moved to the bedroom	milk	garden (0.111)	took (0.011)
sandra went back to the kitchen	john	kitchen (0.501)	dropped (0.027)
mary travelled to the hallway	mary	garden (0.442)	took (0.034)
john got the football there	sandra	hallway (0.394)	kitchen (0.121)
john went to the hallway	daniel	hallway (0.689)	to (0.076)
john put down the football	bedroom	hallway (0.367)	dropped (0.075)
mary went to the garden	kitchen	kitchen (0.483)	daniel (0.029)
john went to the kitchen	garden	garden (0.281)	where (0.026)
sandra travelled to the hallway	hallway	hallway (0.475)	left (0.060)
daniel went to the hallway			
mary discarded the milk			
where is the milk ?			
answer: garden			


Unsupervised Learning

Energy-Based Unsupervised Learning

- **Learning an energy function (or contrast function) that takes**
  - Low values on the data manifold
  - Higher values everywhere else

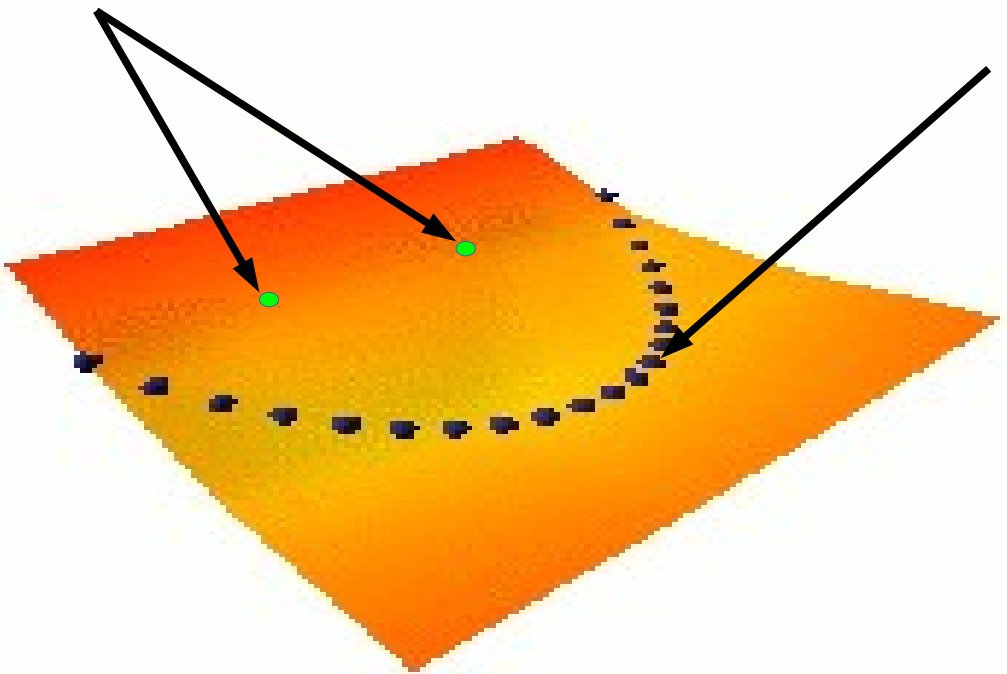




Energy-Based Unsupervised Learning

- **Energy Function:** Takes low value on data manifold, higher values everywhere else
- Push down on the energy of desired outputs. Push up on everything else.
- **But how do we choose where to push up?**

Implausible futures (high energy)

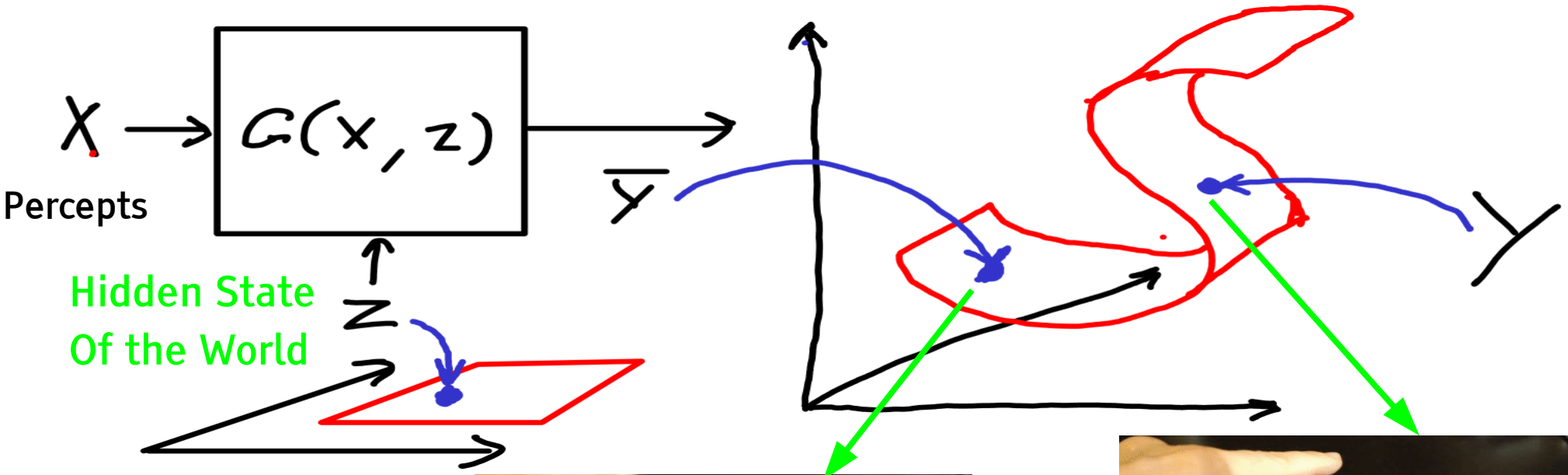


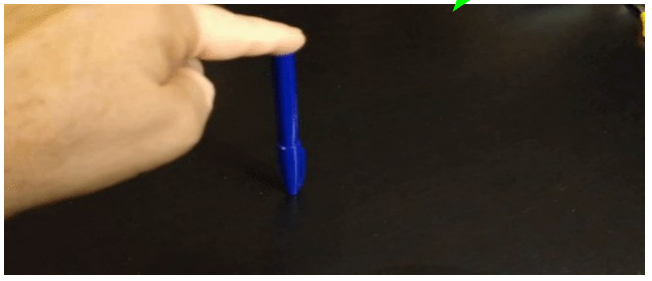
Plausible futures (low energy)

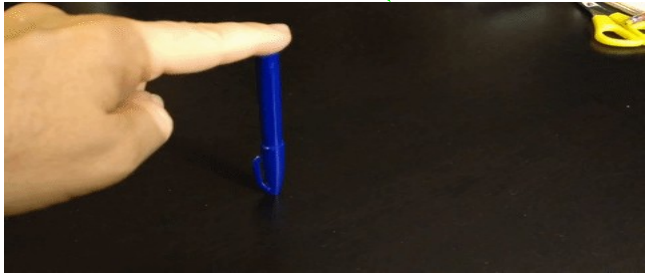

Adversarial Training

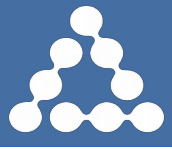
The Hard Part: Prediction Under Uncertainty

- **Invariant prediction: The training samples are merely representatives of a whole set of possible outputs (e.g. a manifold of outputs).**





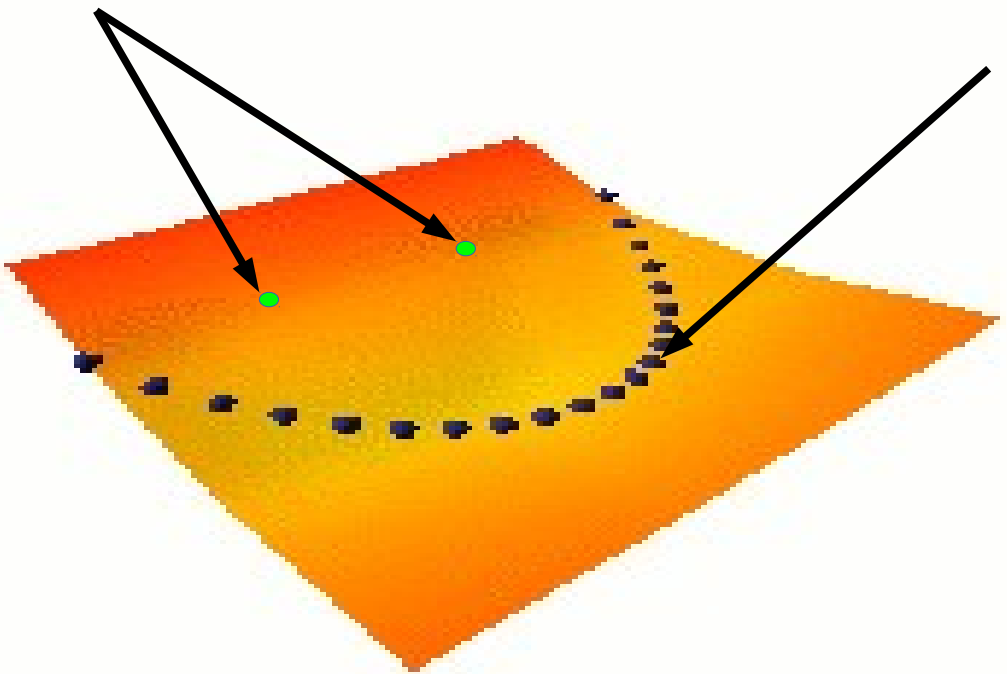




Energy-Based Unsupervised Learning

- **Energy Function:** Takes low value on data manifold, higher values everywhere else
- Push down on the energy of desired outputs. Push up on everything else.
- **But how do we choose where to push up?**

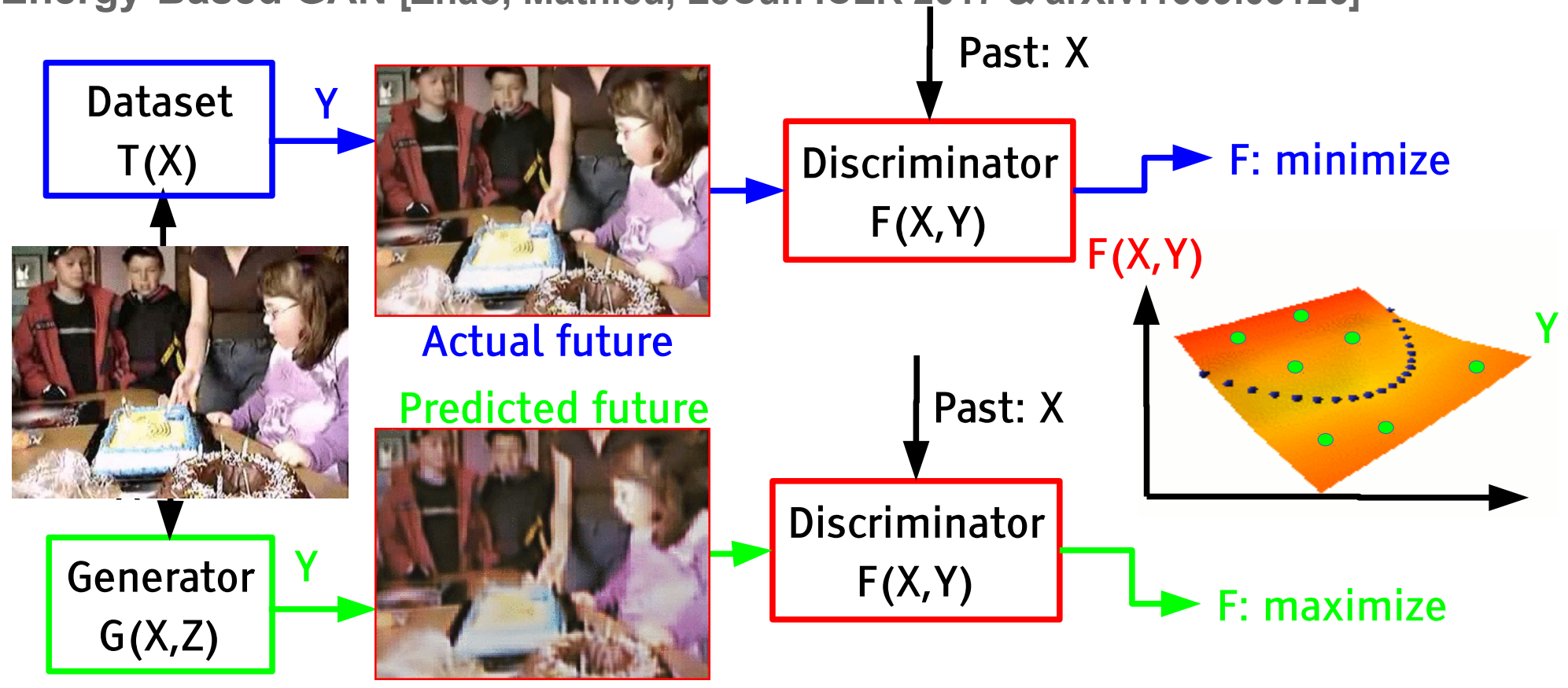
Implausible futures (high energy)

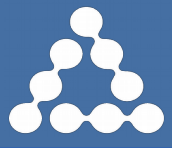


Plausible futures (low energy)

Adversarial Training:
the key to predicting under uncertainty

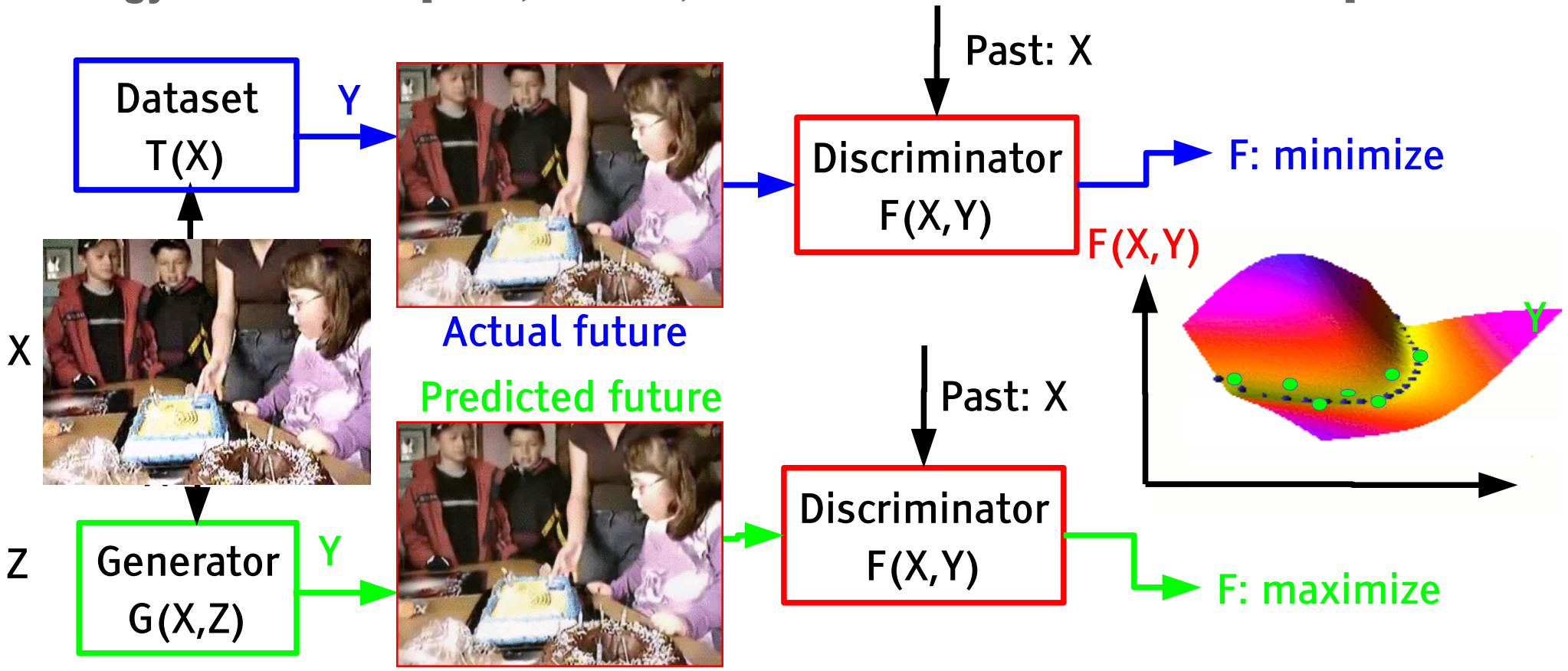
- Generative Adversarial Networks (GAN) [Goodfellow et al. NIPS 2014],
- Energy-Based GAN [Zhao, Mathieu, LeCun ICLR 2017 & arXiv:1609.03126]





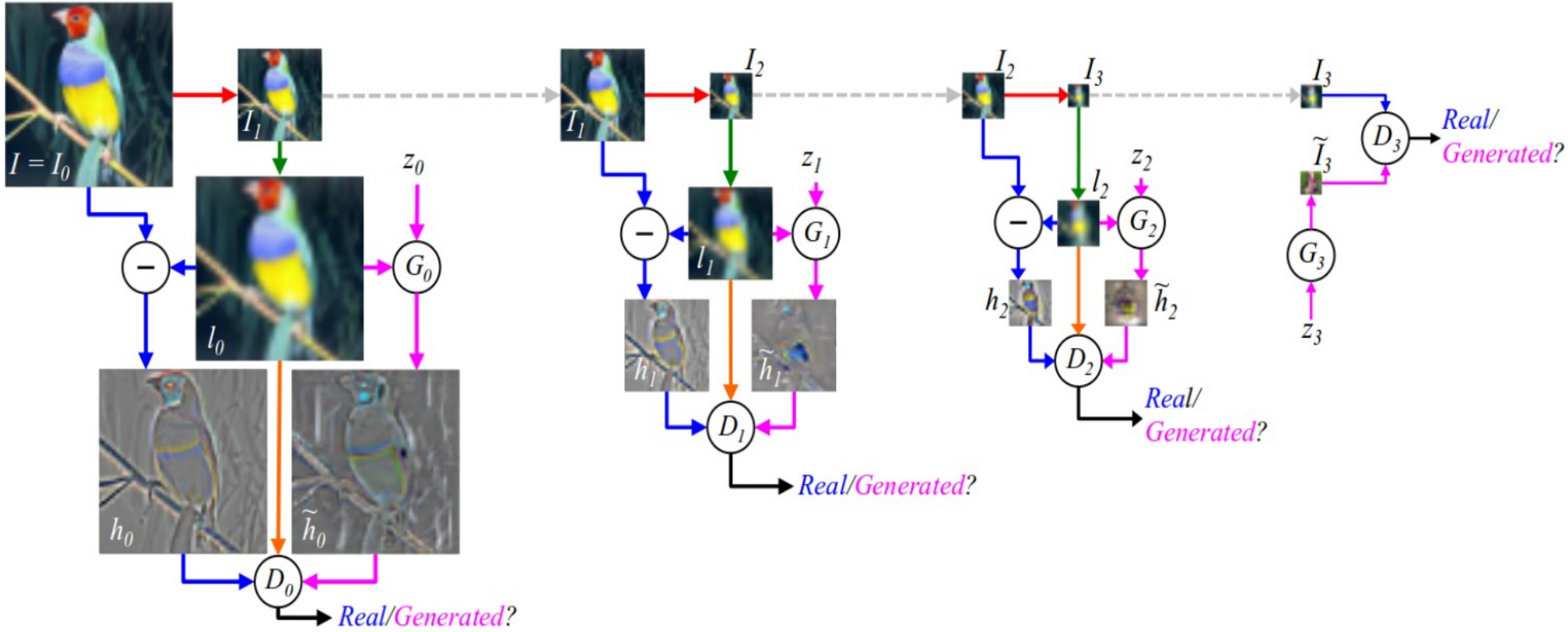
Adversarial Training:
The discriminator is a trainable objective function

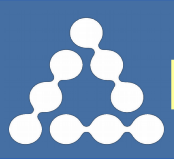
- Generative Adversarial Networks (GAN) [Goodfellow et al. NIPS 2014],
- Energy-Based GAN [Zhao, Mathieu, LeCun ICLR 2017 & arXiv:1609.03126]



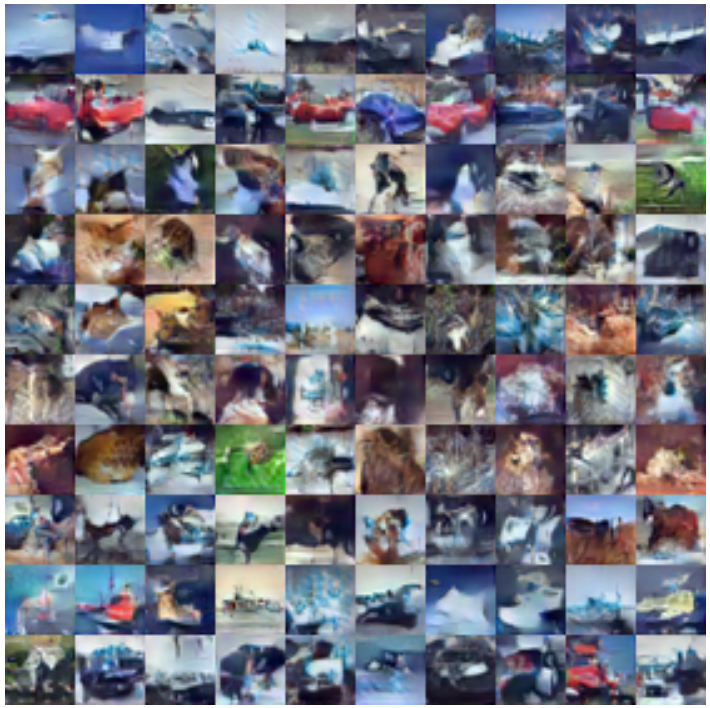
LAPGAN (Laplacian GAN)

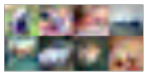
- [Denton et al, NIPS 2015]
- Generator net produces coefficients of a Laplacian Pyramid representation of the image





LAPGAN (Laplacian GAN)

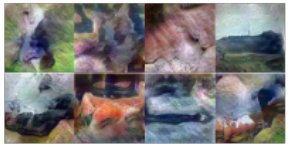




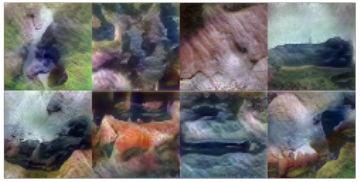
CIFAR-8

CIFAR-16

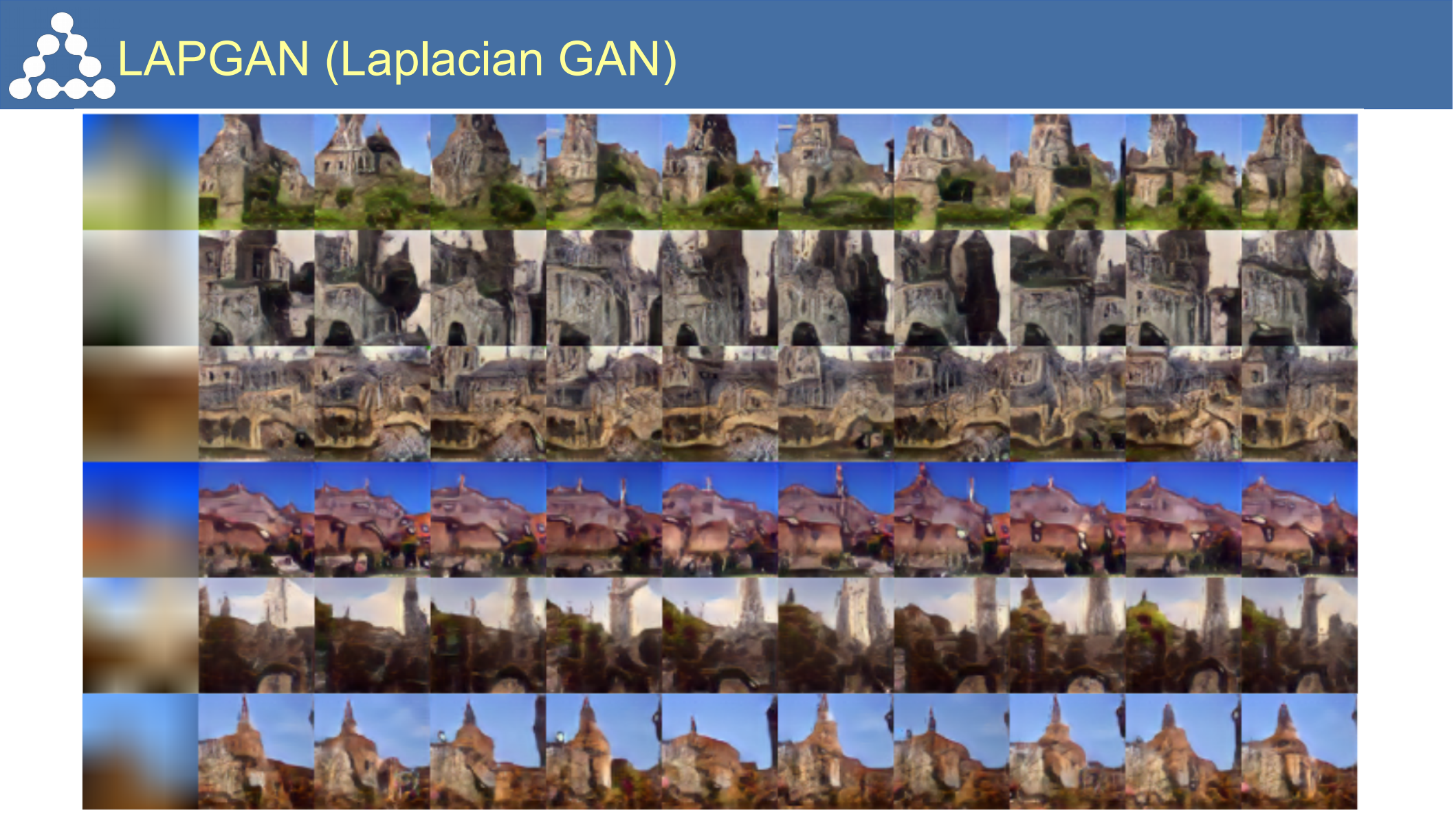
Imagenet-32



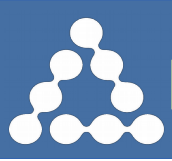
Imagenet-32
(recursive)



Imagenet-32
(recursive)



LAPGAN (Laplacian GAN)



DCGAN: “reverse” ConvNet maps random vectors to images

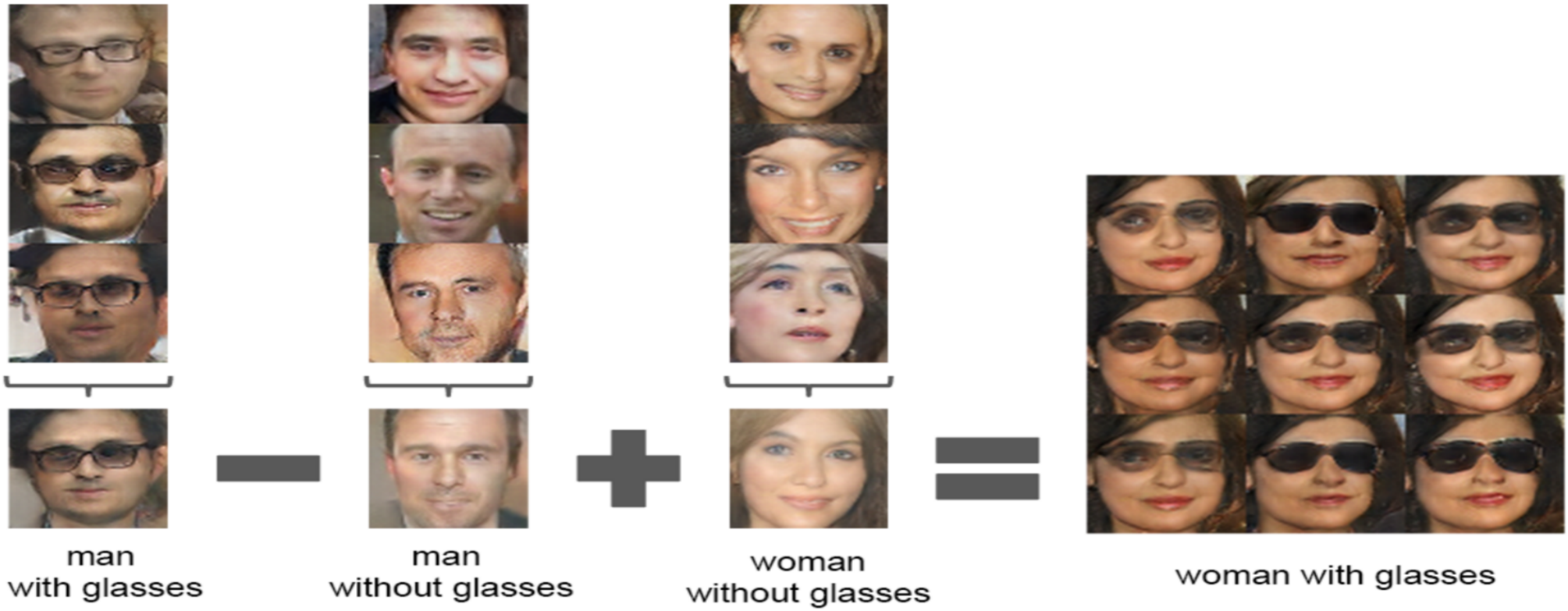
- **DCGAN: adversarial training to generate images.**
- **[Radford, Metz, Chintala 2015]**
  - Input: random numbers; output: bedrooms.

Navigating the Manifold

- DCGAN: adversarial training to generate images.
- Trained on Manga characters
- Interpolates between characters

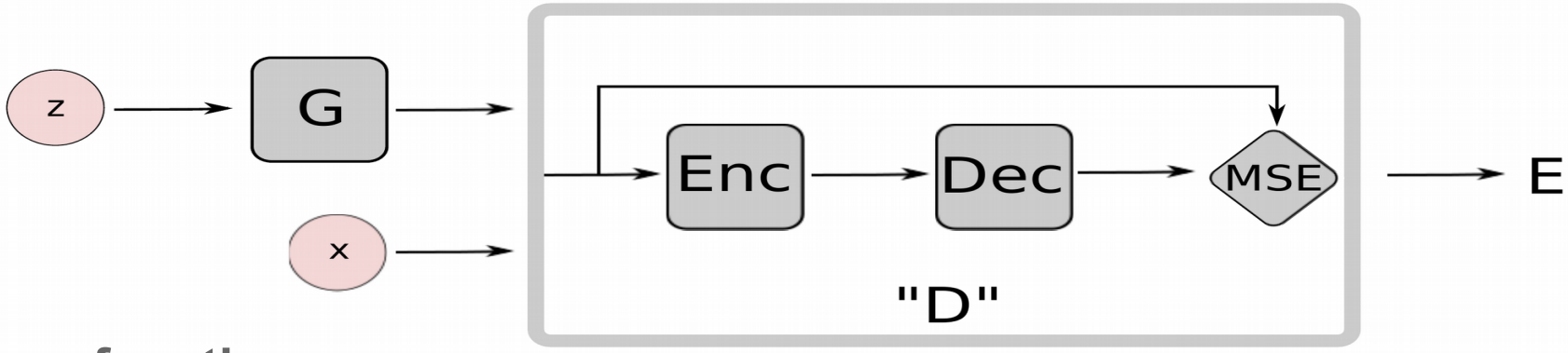
Face Algebra (in DCGAN space)

- **DCGAN: adversarial training to generate images.**
  - [Radford, Metz, Chintala 2015]



Energy-Based GAN [Zhao, Mathieu, LeCun: arXiv:1609.03.126]

- **Architecture: discriminator is an auto-encoder**



- **Loss functions**

$$f_D(x,z) = D(x) + [m - D(G(z))]^+$$
$$= \|Dec(Enc(x)) - x\| + [m - \|Dec\Big(Enc(G(z))\Big) - G(z)\|]^+,$$

$$f_G(z) = \|D(G(z))\|$$
$$= \|Dec\Big(Enc(G(z))\Big) - G(z)\|$$

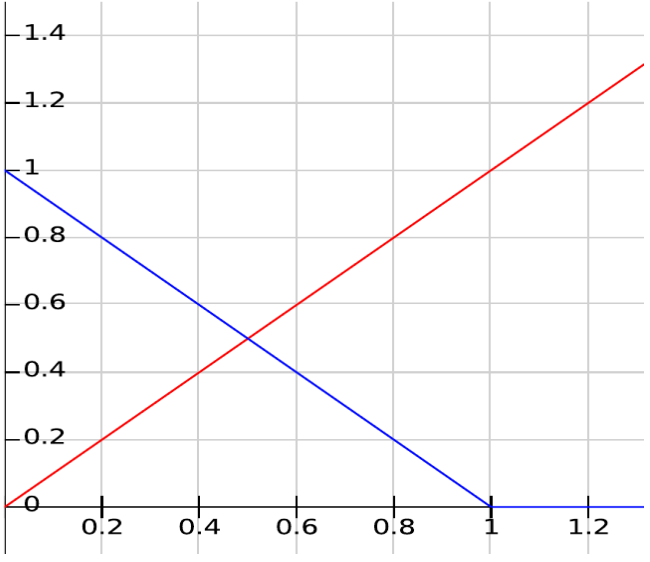
EBGAN Loss function

- **Loss functions for Discriminator and Generator.** Assume D(x) is positive.

$$L_D(x,z) = f(D(x)) + f([m - D(G(z))]^+)$$

$$L_G(z) = f(D(G(z)))$$

- **f must be strictly increasing & convex, with f(0)=0**
  - Examples: half-wave rectification, square



EBGAN solutions are Nash Equilibria

- **Loss functions for Discriminator and Generator. D(x) is positive.**

$$L_D(x,z) = f(D(x)) + f([m - D(G(z))]^+)$$

$$L_G(z) = f(D(G(z)))$$

- **f must be strictly increasing & convex with f(0)=0**
- **(1) there is a Nash equilibrium, (2) if it is reached, the distributions are equal**

We define $V(G,D) = \int_{x,z} \mathcal{L}_D(x,z) p_{data}(x) p_z(z) \mathrm{d}x \mathrm{d}z$ and $U(G,D) = \int_z \mathcal{L}_G(z) p_z(z) \mathrm{d}z$.

$$V(G^*, D^*) \leq V(G^*, D) \qquad \forall D$$
$$U(G^*, D^*) \leq U(G, D^*) \qquad \forall G$$

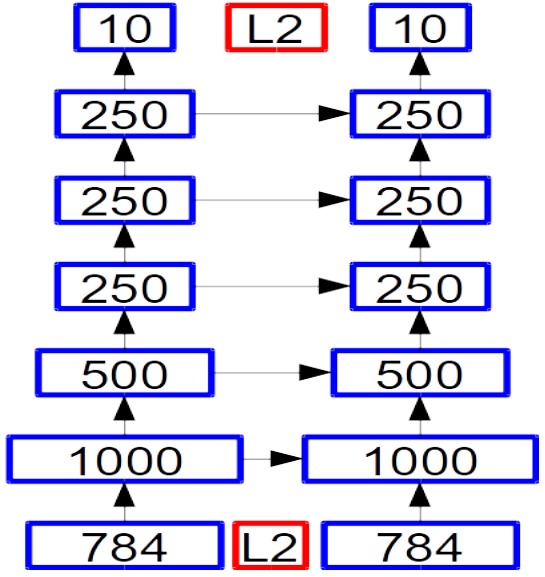
Theorem 1. *If (D^*, G^*) is a Nash equilibrium of the system, then $p_{G^*} = p_{data}$ almost everywhere, and $V(D^*, G^*) = m$.*

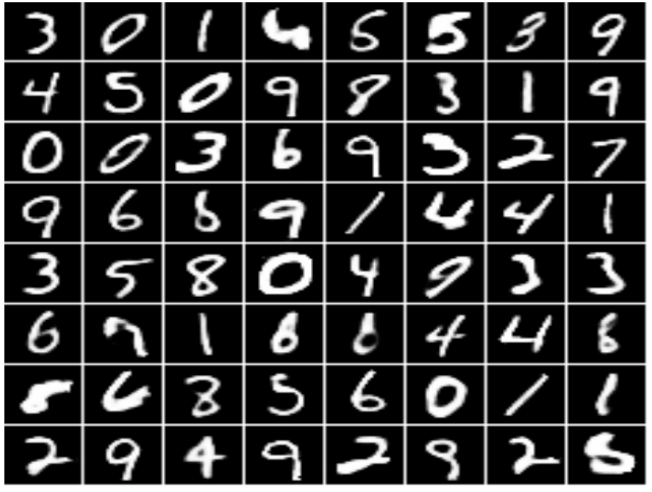
Theorem 2. *Nash equilibrium of this system exists and is characterized by (a) $p_{G^*} = p_{data}$ (almost everywhere) and (b) there exists a constant $\gamma \in [0, m]$ such that $D^*(x) = \gamma$ (almost everywhere).*[1]

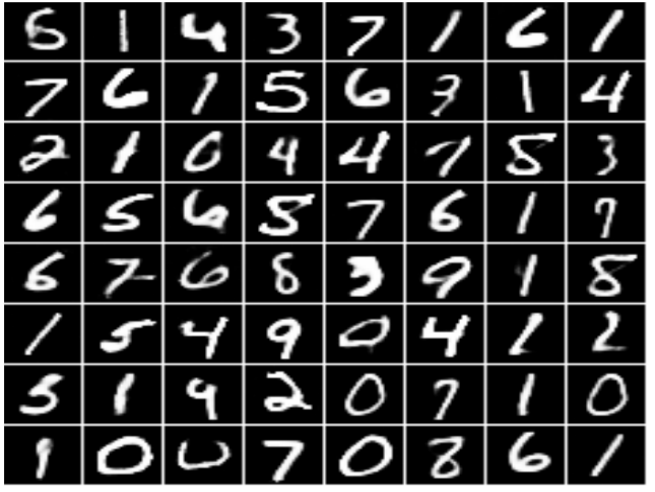
EBGAN in which D(Y) is a Ladder Network

- Ladder Network: auto-encoder with skip connections [Rasmus et al 2015]
- 0.89% error with 1000 samples/class on MNIST with a fully connected nets.

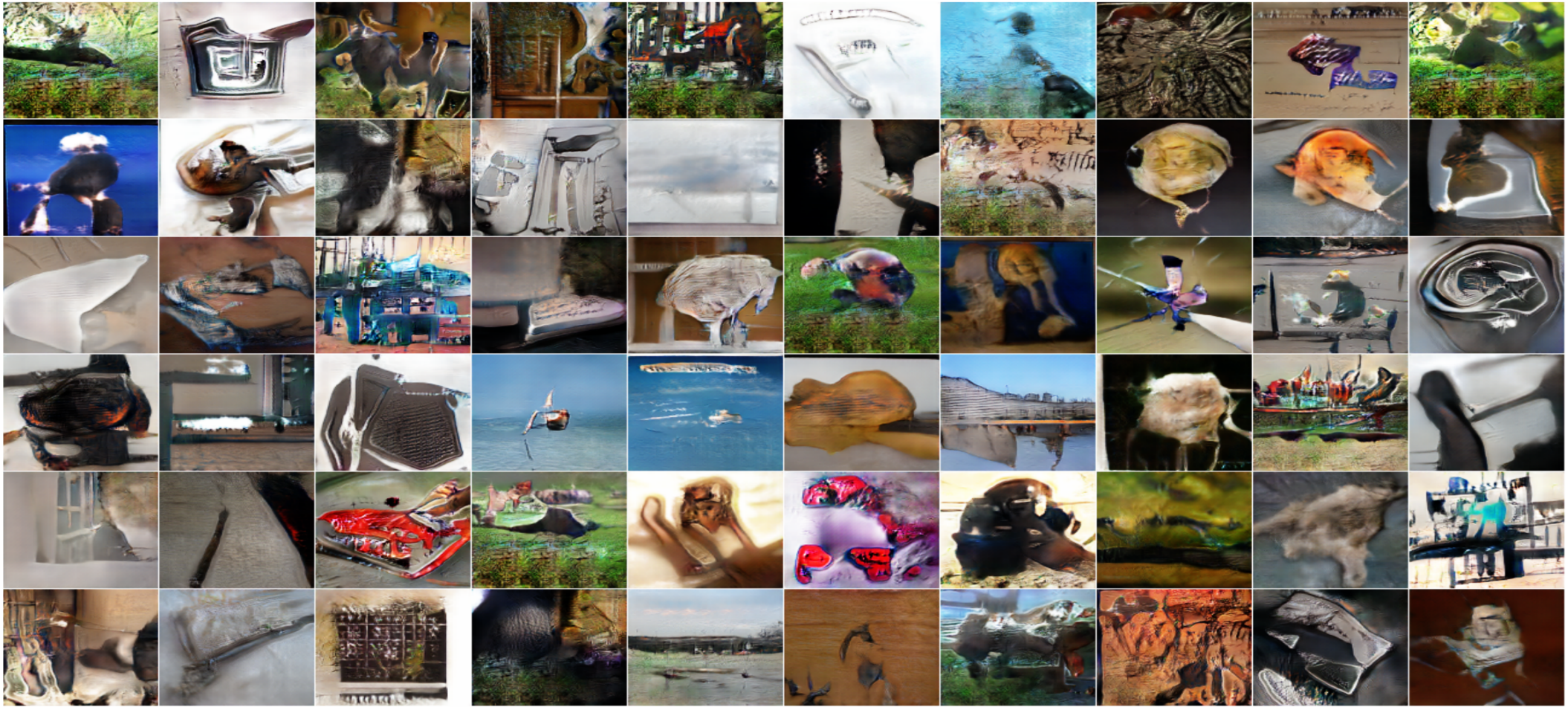
model	100	200	1000
LN bottom-layer-cost, reported in Pezeshki et al. (2015)	1.69±0.18	-	1.05±0.02
LN bottom-layer-cost, reported in Rasmus et al. (2015)	1.09±0.32	-	0.90±0.05
LN bottom-layer-cost, reproduced in this work (see appendix D)	1.36±0.21	1.24±0.09	1.04±0.06
LN bottom-layer-cost within EBGAN framework	**1.04±0.12**	**0.99±0.12**	**0.89±0.04**
Relative percentage improvement	23.5%	20.2%	14.4%

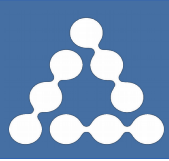






Energy-Based GAN trained on ImageNet at 128x128 pixels





Energy-Based GAN trained on ImageNet at 256x256 pixels

- **Trained on dogs**

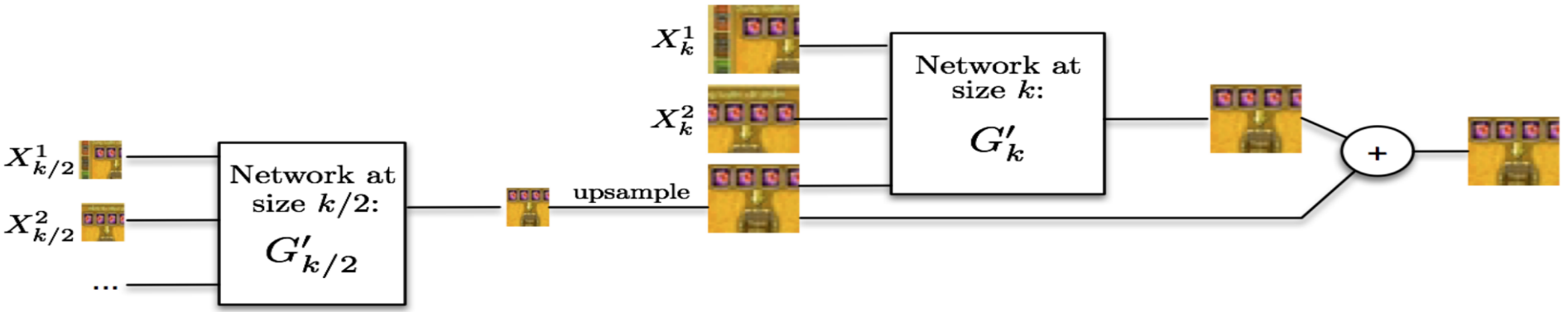
Video Prediction
(with adversarial training)

[Mathieu, Couprie, LeCun ICLR 2016]
arXiv:1511:05440

Multi-Scale ConvNet for Video Prediction

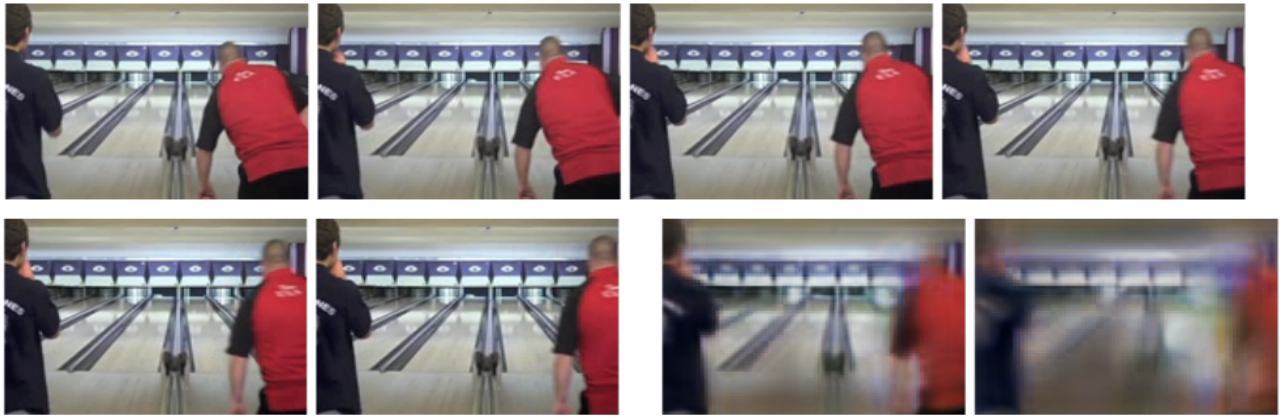
- 4 to 8 frames input → ConvNet → 1 to 8 frames out
- Multi-scale ConvNet, without pooling
- If trained with least square: **blurry output**

Predictor (multiscale ConvNet Encoder-Decoder)

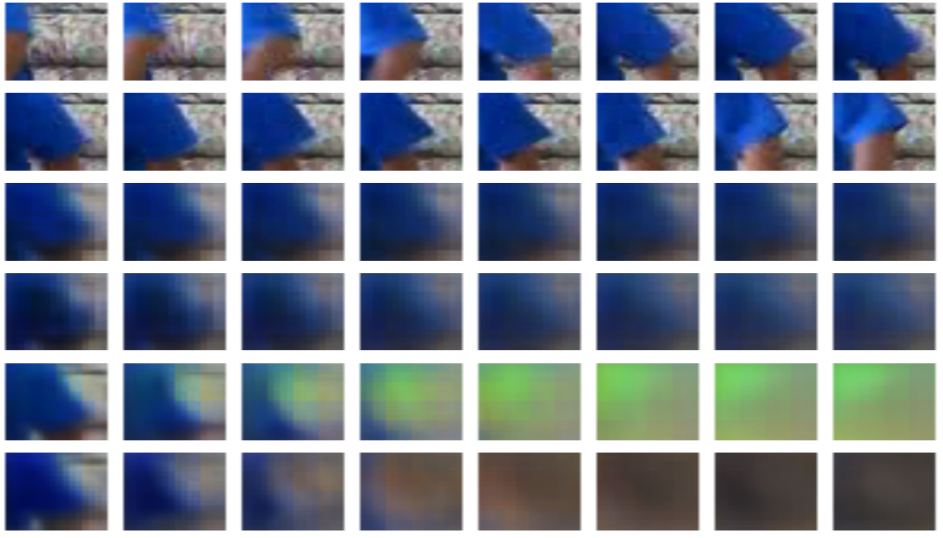


Can't Use Squared Error: blurry predictions

- The world is unpredictable
- MSE training predicts the average of possible futures:
- blurry images.



Ground truth ℓ_2 result



Input

Ground truth

ℓ_1

ℓ_2

ℓ_1 recursive

ℓ_2 recursive

Multi-Scale ConvNet for Video Prediction

Architectures

Models 4 frames in input – 1 frame in output

Generative network scales	G_1	G_2	G_3	G_4
Number of feature maps	128, 256, 128	128, 256, 128	128, 256, 512, 256, 128	128, 256, 512, 256, 128
Conv. kernel size	3, 3, 3, 3	5, 3, 3, 5	5, 3, 3, 3, 5	7, 5, 5, 5, 5, 7
Adversarial network scales	D_1	D_2	D_3	D_4
Number of feature maps	64	64, 128, 128	128, 256, 256	128, 256, 512, 128
Conv. kernel size (no padding)	3	3, 3, 3	5, 5, 5	7, 7, 5, 5
Fully connected	512, 256	1024, 512	1024, 512	1024, 512

Models 8 frames in input – 8 frames in output

Generative network scales	G_1	G_2	G_3	G_4
Number of feature maps	16, 32, 64	16, 32, 64	32, 64, 128	32, 64, 128, 128
Conv. kernel size	3, 3, 3, 3	5, 3, 3, 3	5, 5, 5, 5	7, 5, 5, 5, 5
Adversarial network scales	D_1	D_2	D_3	D_4
Number of feature maps	16	16, 32, 32	32, 64, 64	32, 64, 128, 128
Conv. kernel size (no padding)	3	3, 3, 3	5, 5, 5	7, 7, 5, 5
Fully connected	128, 64	256, 128	256, 128	256, 128

Multi-Scale ConvNet for Video Prediction

Results on UCF101 (10% of test images)

- 8 frames input → 8 frames output

	1st frame prediction scores		8th frame prediction scores	
	PSNR	Sharpness	PSNR	Sharpness
ℓ_2	18.3	0.47	16.4	0.36
Adv	21.0	0.65	18.9	0.54
ℓ_1	21.2	0.57	18.3	0.51
GDL ℓ_1	**21.8**	**0.87**	**19.2**	**0.79**

Results on UCF101 (10% of test images)

- 4 frames input → 1 frames output

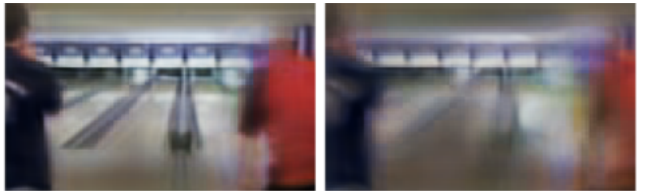
	1st frame prediction scores		2nd frame prediction scores	
	PSNR	Sharpness	PSNR	Sharpness
ℓ_2	20.1	0.48	14.1	0.29
ℓ_1	22.2	0.58	16.0	0.48
GDL ℓ_1	23.9	0.69	18.5	0.60
Adv	24.4	**0.95**	18.9	**1.00**
Adv+GDL	**27.2**	0.77	**22.6**	0.68

Multi-Scale ConvNet for Video Prediction

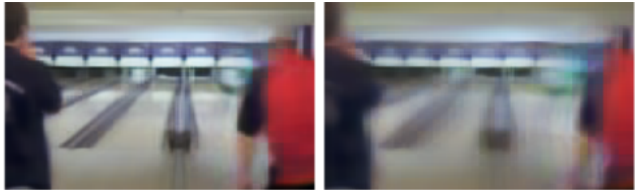
Examples

Input frames

Ground truth



ℓ_2 result



ℓ_1 result

GDL ℓ_1 result

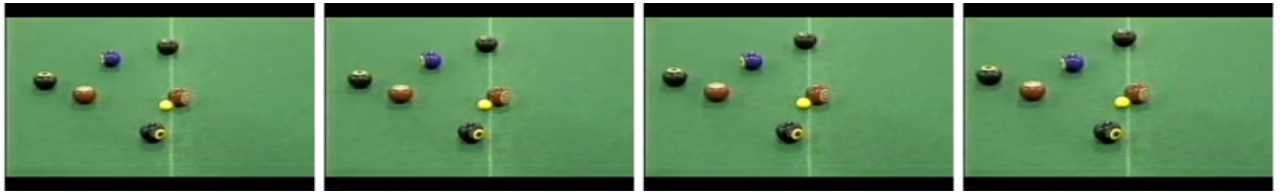
Adversarial result

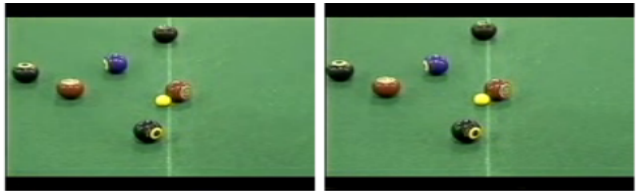
Adversarial+GDL result

Multi-Scale ConvNet for Video Prediction

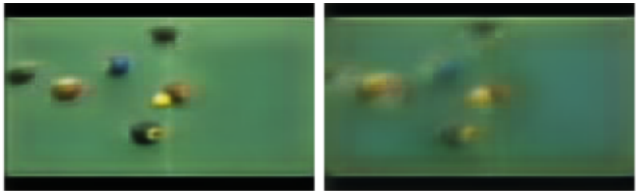
Examples

Input frames

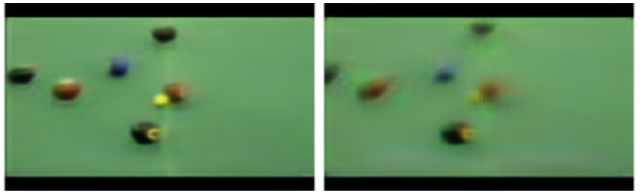




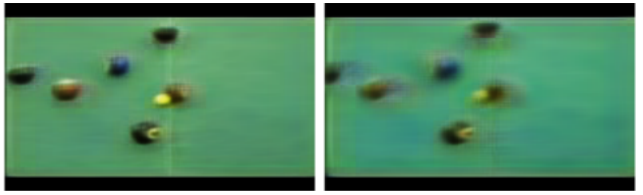
Ground truth



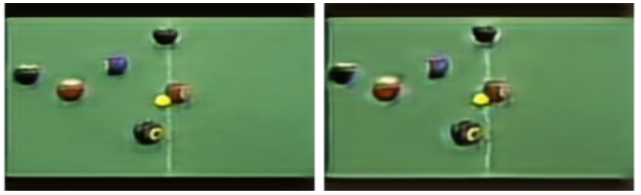
ℓ_2 result



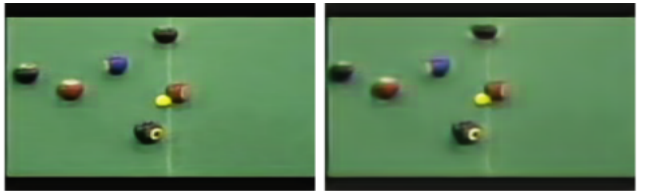
ℓ_1 result



GDL ℓ_1 result



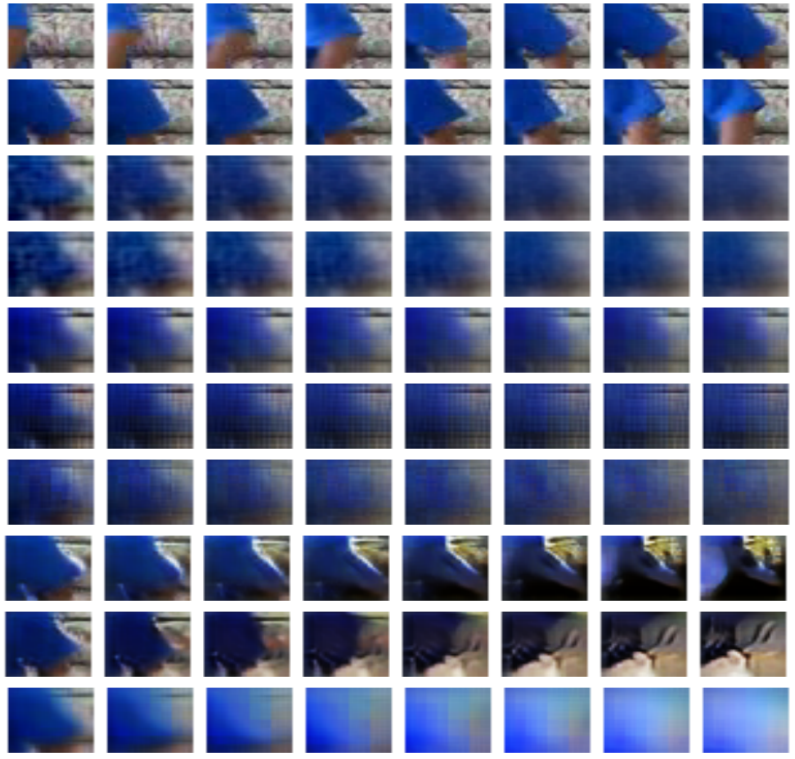
Adversarial result



Adversarial+GDL result

Multi-Scale ConvNet for Video Prediction

Comparison with [Srivastava et al. 2015] who used LSTM.



Input

Ground truth

LSTM 2048

LSTM 4096

GDL ℓ_1

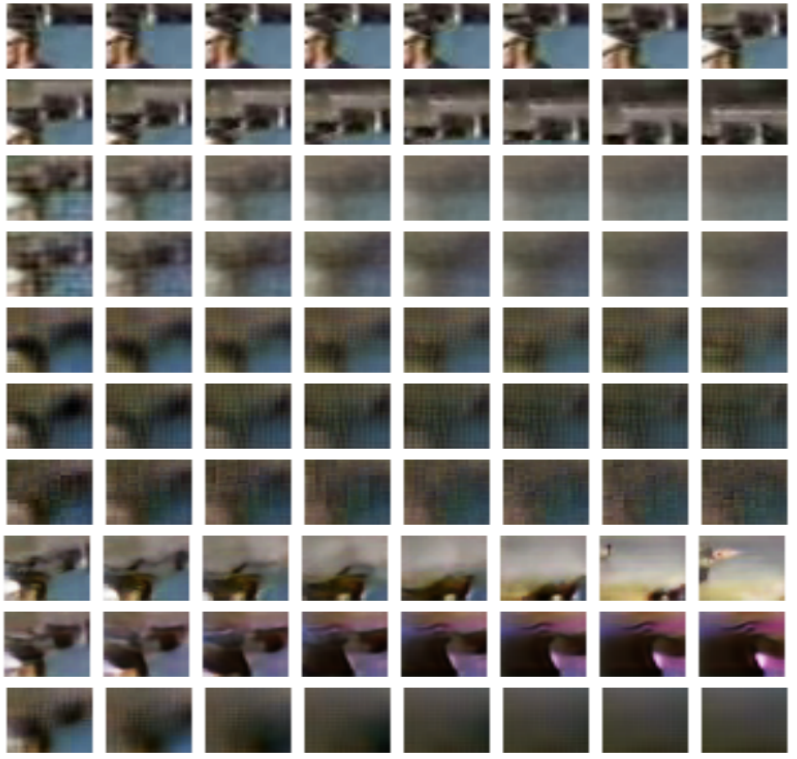
GDL ℓ_2

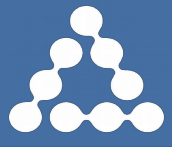
Adversarial

Adv. recursive

Adv. rec. + GDL

GDL ℓ_1 recursive





Predictive Unsupervised Learning

- Our brains are “prediction machines”
- Can we train machines to predict the future?
- Some success with “adversarial training”
  - [Mathieu, Couprie, LeCun arXiv:1511:05440]
- But we are far from a complete solution.

Video Prediction: predicting 5 frames

Video Prediction: predicting 5 frames

Video Prediction: predicting 50 frames

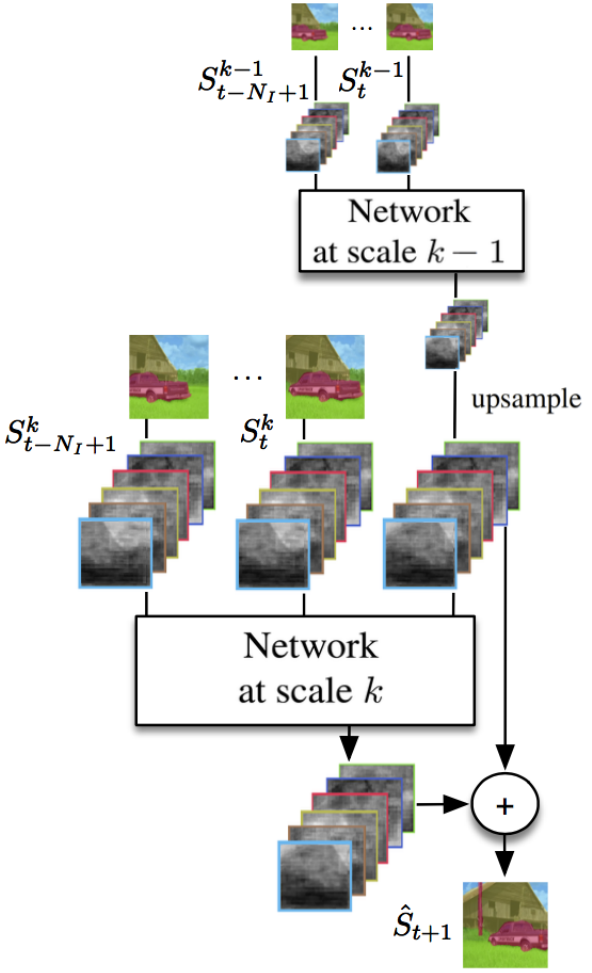
Video Prediction of semantic segmentation

[Neverova, Luc, Couprie, Verbeek, LeCun 2017]

Temporal Predictions of Semantic Segmentations

- **Predictions a single future frame**
  - CityScape dataset [Cordt et al. CVPR 2016]

Method	PSNR	SSIM	IoU GT	IoU SEG	IoU-MO GT	IoU-MO SEG
Copy last input	20.6	0.65	49.4	54.6	43.4	48.2
Warp last input	20.9	0.67	50.4	55.5	44.9	49.8
Model X2X	24.0	**0.77**	23.0	22.3	12.8	11.4
Model S2S	—	—	**58.3**	64.9	53.8	59.8
Model S2S-adv.	—	—	**58.3**	**65.0**	**53.9**	**60.2**
Model XS2X	**24.2**	**0.77**	22.4	22.5	10.8	10.0
Model XS2S	—	—	58.2	64.6	53.7	59.9
Model XS2XS	24.0	0.76	55.5	61.1	50.7	55.8



Temporal Predictions of Semantic Segmentations

- Prediction 9 frames ahead (0.5 seconds)
- Auto-regressive model


X_t, S_t


X_{t+9}, GT


Batch predictions at $t+3$


at $t+9$


Autoregressive pred. at $t+3$


at $t+9$


AR fine-tune pred. at $t+3$


at $t+9$


X_t, S_t


X_{t+9}, GT


Optical flow at $t+3$

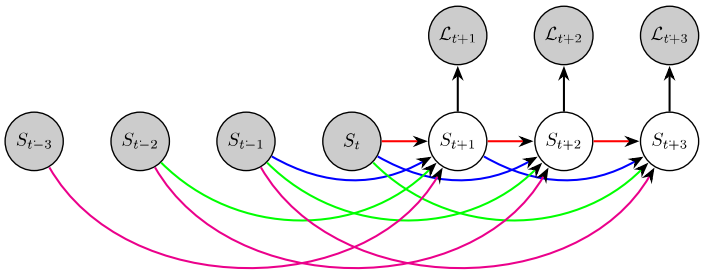

at $t+9$


Autor. adv. pred. at $t+3$


at $t+9$


AR fine-tune pred. at $t+3$

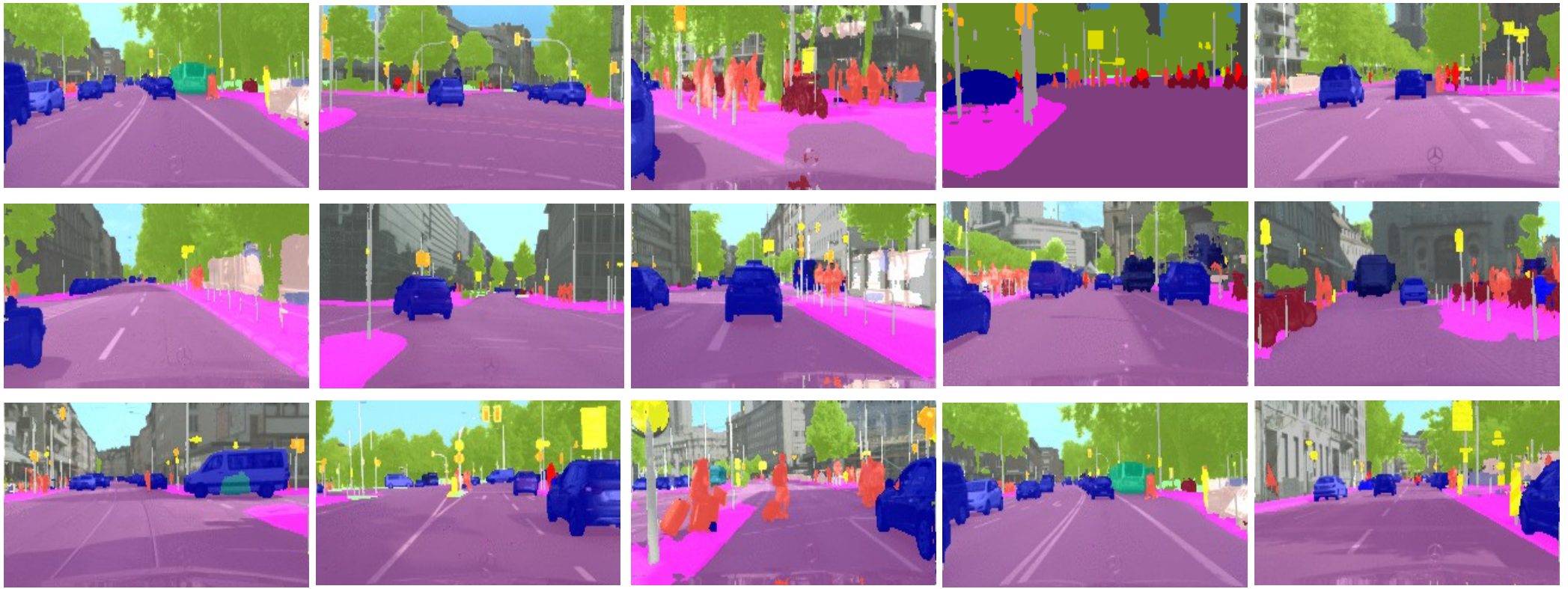

at $t+9$



Model	IoU GT	IoU SEG	IoU-MO GT
Copy last input	36.9	39.2	26.8
Warp last input	37.5	39.5	27.9
S2S, AR	45.3	47.2	36.4
S2S-adv, AR	45.1	47.2	37.3
S2S, AR, fine-tune	**46.7**	**49.7**	**39.3**
XS2XS, AR	39.3	40.8	27.4
S2S, batch	42.1	44.2	32.8
XS2S, batch	42.3	44.6	33.1
XS2XS, batch	41.2	43.5	31.4

Temporal Predictions of Semantic Segmentations

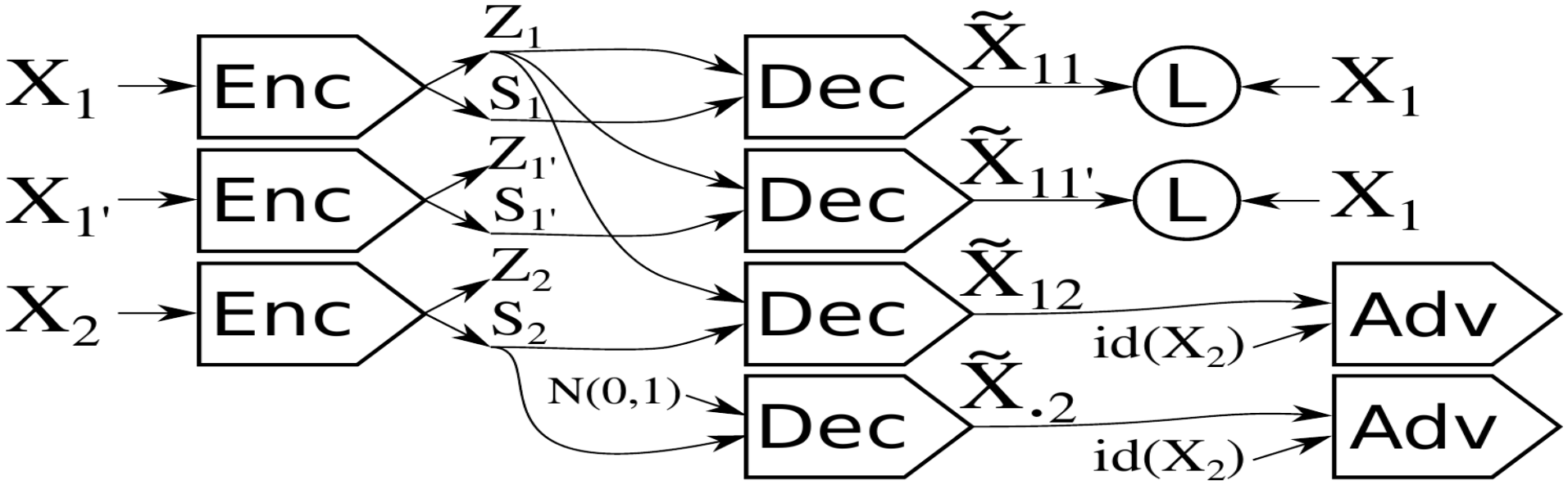
- Prediction 9 frames ahead (0.5 seconds)
- Auto-regressive model




Style Transfer
(Mathieu et al. NIPS 2016)

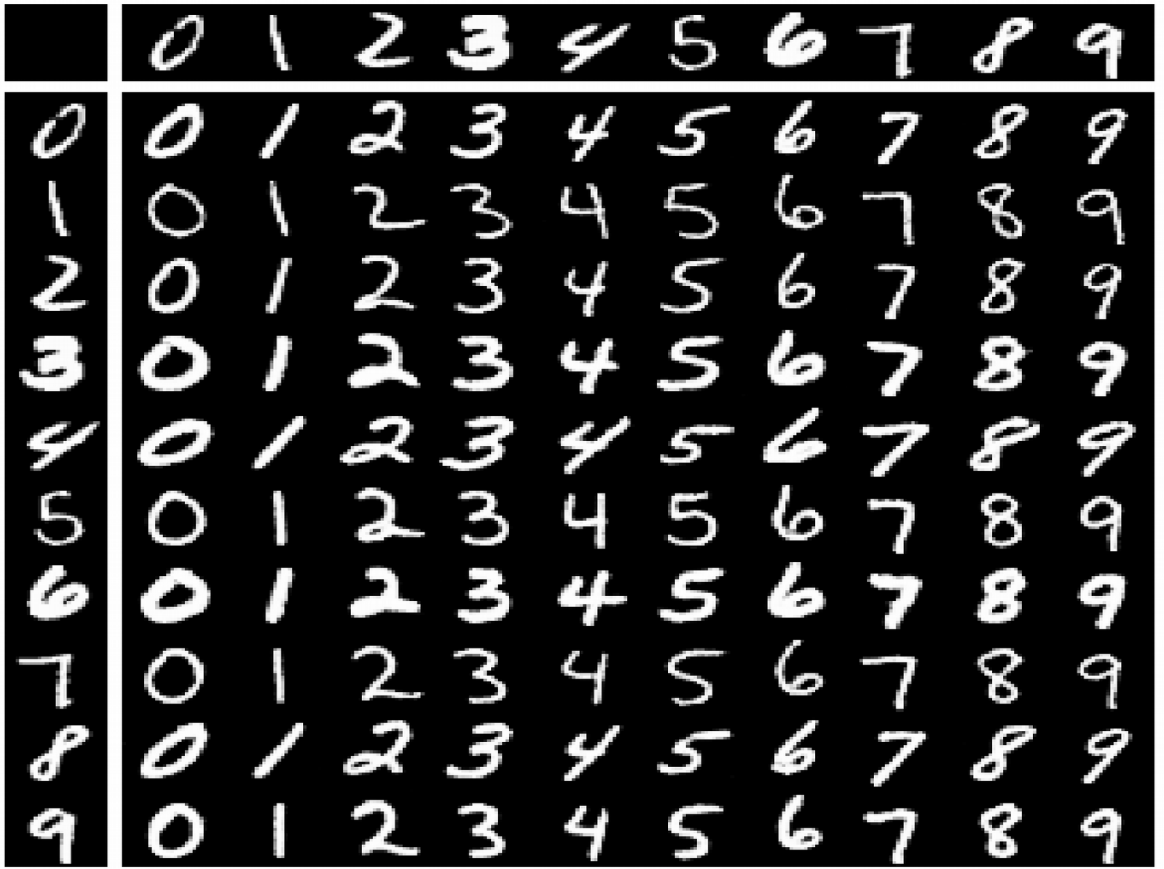
Style transfer architecture

- X1 and X1' have same "label" (or known features)
- X2 can have any label
- S1 and S1' are meant to represent the "label" (the known part of the representation)
- Z1, Z1' and Z2 are the unspecified part (eg the pose)



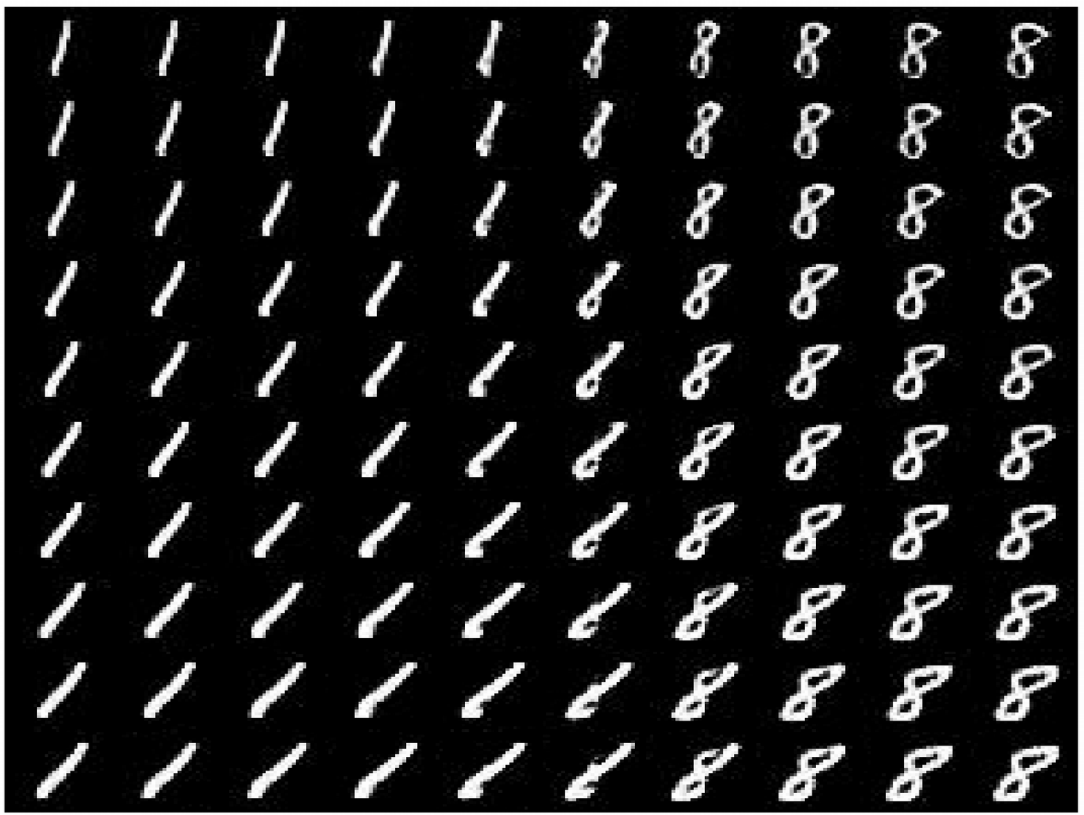
Style transfer results

- Transfer category from top row to style of left column



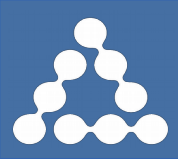
Style transfer: interpolation

- Interpolate between top left and bottom right characters
- Style changes vertically, identity changes horizontally.



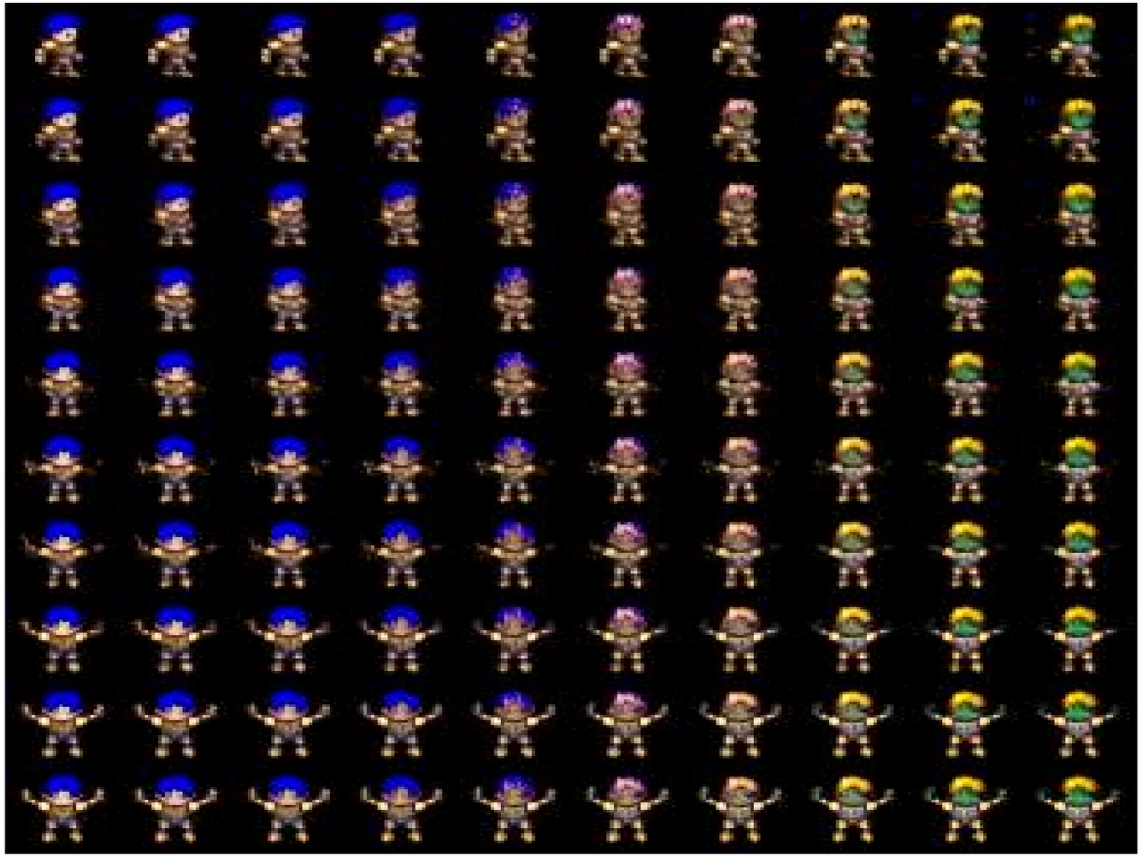
Style transfer results

- Transfer category from top row to style of left column



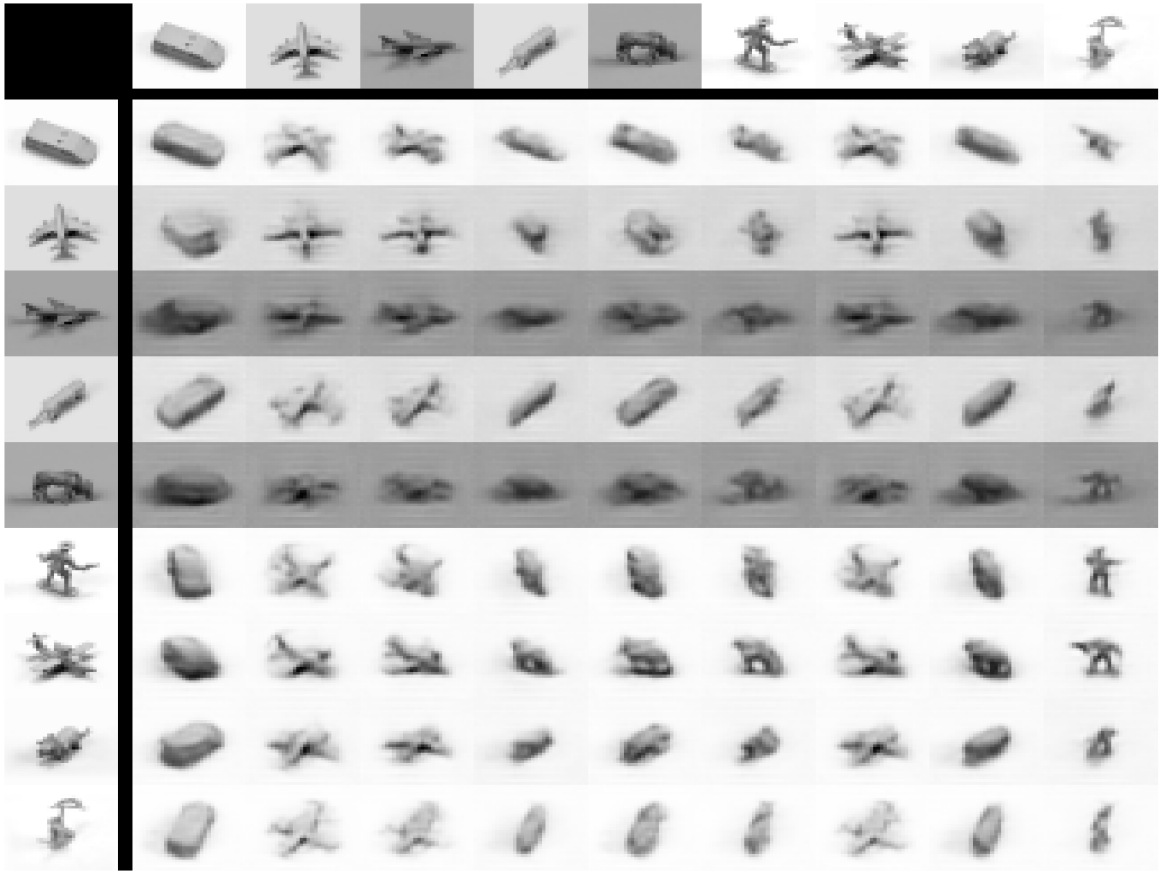
Style transfer: interpolation

- Interpolate between top left and bottom right characters
- Style changes vertically, identity changes horizontally.



Pose transfer results

- Transfer category from top row to orientation of left column



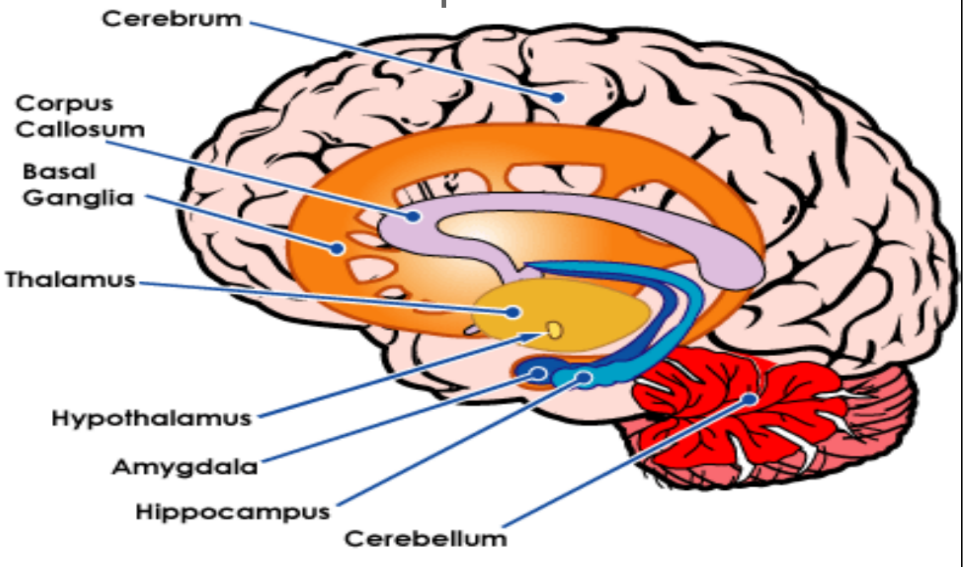
Pose transfer results

- Transfer category from top row to orientation of left column

Can an AI system's Objective Function be trainable? Imitation Learning & Inverse Reinforcement Learning

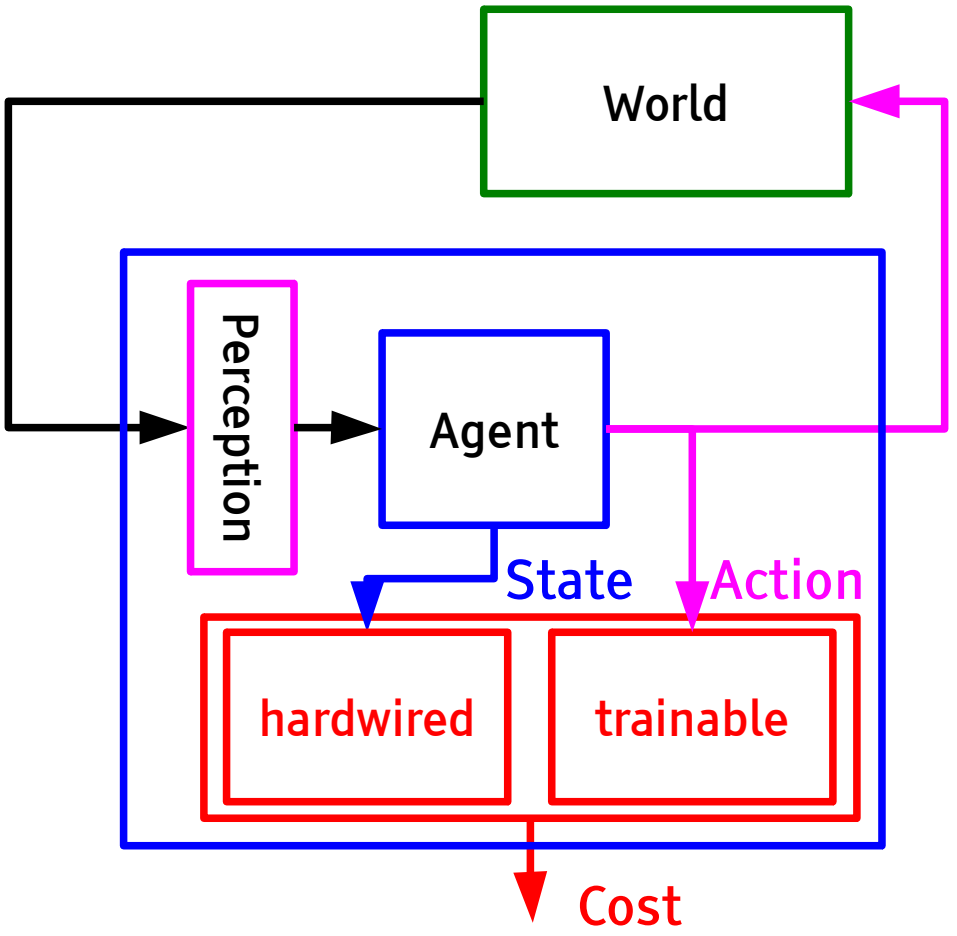
What will future AI systems be like?

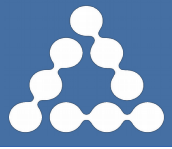
- **Human and animal behavior has basic "drives" hardwired by evolution**
  - Fight/flight, hunger, self-preservation, pain avoidance, desire for social interaction, etc…
- **Humans do bad things to each other because of these drives (mostly)**
  - Violence under threat, desire for material resource and social power…
- **But an AI system will not have these drives unless we build them into it.**
- **It's difficult for us humans to imagine an intelligent entity without these drives**
  - But they are specific to humans
  - We have plenty of different forms of intelligence in the animal world



Aligning the Objective with Human Behavior

- **Make the objective have two components**
- **1. A hardwired, immutable "safeguard" objective**
  - Call it the instinct
- **2. A Trainable objective that estimates the value function of its human trainers**
  - It's trained through adversarial training / Inverse RL.





Inverse RL through Adversarial Training

- Train the objective to emulate the (unknown) objectives of the human trainers
- Objective trains to distinguish trainer actions from agent actions
- Agent plans/trains to minimize objective